INFORMATICA, 2014, Vol. 25, No. 4, 581–616
© 2014 *Vilnius University*
DOI: http://dx.doi.org/10.15388/Informatica.2014.31

Advanced Planning and Scheduling Systems: Modeling and Implementation Challenges

Audrone LUPEIKIENE[1] *, Gintautas DZEMYDA[1], Ferenc KISS[2]
Albertas CAPLINSKAS[1]
[1] *Institute of Mathematics and Informatics, Vilnius University*
Akademijos 4, LT-08663 Vilnius, Lithuania
[2] *INFOTA Research Institute of the Foundation for Information Society*
1507 Budapest, PO Box 213, Hungary
e-mail: audrone.lupeikiene@mii.vu.lt, gintautas.dzemyda@mii.vu.lt, kiss.ferenc@infota.org, albertas.caplinskas@mii.vu.lt

Received: March 2014; accepted: October 2014

Abstract. The paper summarizes the results of research on the modeling and implementation of advanced planning and scheduling (APS) systems done in recent twenty years. It discusses the concept of APS system – how it is thought of today – and highlights the modeling and implementation challenges with which the developers of such systems should cope. Some from these challenges were identified as a result of the study of scientific literature, others – through an in-depth analysis of the experience gained during the development of real-world APS system – a Production Efficiency Navigator (PEN system). The paper contributes to APS systems theory by proposing the concept of an ensemble of collaborating algorithms.

Key words: advanced planning and scheduling, optimization, simulation, reference modeling, architectural modeling.

1. Introduction

Advanced planning and scheduling (APS) systems already have a long, more than twenty years history. A plenty of research papers discussing the various aspects of APS systems were published in these years. However, it seems that none of them concentrates on the modeling and implementation challenges in APS systems. The questions "What is the nature of these challenges?", "How many challenges and to which extent were answered?", "What are the modeling and implementation problems that remain unsolved?" still have no exhaustive answers grounded on a systematic investigation of these questions. This paper summarizes the research done in this field, discusses the concept of APS system from today's point of view, aims to answer some among above listed questions and contributes to APS systems theory by proposing the concept of an ensemble of collaborating algorithms.

*Corresponding author.

The results are based on two sources: the analysis of research papers, which investigate APS systems, and the experience of authors gained working in a real-world project called Production Efficiency Navigator (PEN system) for development of a particular APS system.

The PEN system is an APS system, which is oriented to the make-to-order one-plant manufacturing systems and implements a predictive-reactive scheduling approach, which combines both predictive medium-term planning and scheduling, and replanning and rescheduling. It should be used together with some ERP system as an ad-on for this system. The input of the PEN system, when it works in replanning and rescheduling mode, includes: description of the detected exception; actual state of running plan or schedule that should be repaired; and search strategy constraints. Data about the actual state of the plan or the schedule under consideration must be downloaded from the ERP. Constraints on the search strategy (e.g. to search keeping the current deadline, to search keeping the current budget, etc.) must be set when invoking the system. In addition, enterprise map, decision matrix, models library, simulation and optimization procedures library, goals table, management focus, resources and materials tables, and thresholds table must be already stored in the system and accessible for its modules. The materials table describes the acceptable substitutions of the materials that were foreseen in the material requirements plan. The output of the PEN system is a set of repaired plans/schedules (one for each alternative) and their summaries. The repaired plans/schedules can be optimal or near optimal. In some cases the system is not able to absorb the negative impact caused by the occurrence of exception and to repair the plan or schedule. In such cases it generates the empty set of results. It means that full replanning from the scratch must be done by ERP system. The summary of a plan/schedule includes a numerical estimate of solution taking into account the multiple criteria character of the problem caused by the stated optimization goals. For details the reader is referred to Caplinskas *et al.* (2012). PEN System differs from the other similar APS systems in three aspects: (a) it provides specific rule-based mechanisms for production re-scheduling process allowing to take into account knowledge about a specifics of production system of a particular enterprise; (b) it offers special functionality – sandbox functionality – to experiment with the multiple scenarios for special events, risks or failures; (c) it implements the schedule optimization algorithms taking into account business goals and management focuses. The PEN project confirmed that the modeling and implementation challenges discussed in our paper were also incident to this project.

Despite the great amount of quoted research papers this paper does not pretend to be the Systematic Literature Review on the topic. There were chosen only these sources, which, in our opinion, are representative for the discussed approaches and present the significant results.

The remain of this paper is organized as follows: Section 2 analyzes the concept of APS system from today's perspective, Section 3 describes the identified modeling and implementation challenges of APS systems, Section 4 discusses the concept of an ensemble of collaborating algorithms and, finally, Section 5 concludes the paper.

2. Advanced Planning and Scheduling (APS) Systems

2.1. *Evaluation of Current Manufacturing Planning and Control Systems*

The concept of advanced planning and scheduling was developed by join efforts of scientists and software developers as an answer to the challenges posed by limitations of the classical ERP systems. For example, even in advanced ERP systems lot sizing and sequencing are considered as independent decisions, bill of materials typically does not include routing data, it is difficult to maintain the consistency between master production schedule and plant floor level schedule, and to evaluate all the effects on final customer orders caused by occurrences of unpredicted events at plant floor level. ERP schedulers cannot take into account the material constraints, changeover constraints, sub-resource allocation constraints, multi-task constraints and the other complex constraints. They are unable to optimize bottleneck processes (Consortium PSLX, 2005; EyeOn, 2014).

The core of any early ERP system was so-called material requirements planning (MRP) system. Such ERP system can be defined as "*as an integrated system for production planning and control, with level-by-level bill-of-material explosion and netting as its core functionality*" (Peeters, 2009). Such systems prepared production plans in two steps (Consortium PSLX, 2005; Bubenik, 2011; Theeuwen, 2007). In the first step, they calculated material requirements, using forecast or master production schedule as input data and assuming infinite capacity, in the second step – the amount of capacity required. In other words, the materials and capacities were planed separately. Availability of capacities was taken into account very late in the production process. Therefore the plans, prepared using such stepwise procedure, were often not feasible because the system "allows the release of work orders to the shop floor without consideration of component parts availability" (Ptak and Smith, 2008). In order to transform a not feasible plan into executable one, input data should be changed and the whole planning procedure repeated over again. The system did not suggest, what changes in input data should be done. Besides, even in cases when feasible schedules for shop floor activities were prepared, they were not optimized and often were postponed by shop floor control staff in order to optimize operation sequence of work orders (Hopp, 2007).

Later MRP procedure was replaced by MRPII procedure. In this extended procedure material requirement planning was followed by capacity requirements planning, scheduling and other sequentially executed planning procedures (Hopp, 2007). Due to its sequential nature, the planning process did lead to long processing times. Consequently, this causes long planning cycles, which, in turn, cause that final planning results become outdated. Besides, planning and scheduling remains based on unlimited resource availability, MRPII cannot perform cross-plant planning, including logistics and distribution, and has many other shortcomings because at the core of MRPII still remains basic production planning principles developed in 1950's (Ptak and Smith, 2008). Due to the very limited optimization of shop floor schedules, the users should do optimization manually using Excel™, Access™ or even sticky notes and scheduling white boards (Ptak and Smith, 2008).

2.2. *The Concept of APS*

The concept of advanced planning and scheduling was developed in the mid 1990s with the intention to solve the above mentioned problems. However, up to date, there is not generally accepted definition of a term Advanced Planning and Scheduling (APS) System exists. This term is still widely used ambiguous.

Some sources define this term at conceptual level considering APS systems as a new approach to manufacture planning. For example, in Consortium PSLX (2005) an APS system is defined "*as a system and methodology in which decision-making, such as planning and scheduling for industries, is federated and synchronized between different divisions, within or between enterprises, to achieve total and autonomous optimization*". According to this definition, APS system is not a part of ERP, but rather an entire planning and scheduling system within an enterprise. So, planning and scheduling process is thought of as a primary aspect of decision making in manufacturing enterprises. More exactly, decision-making is perceived as a process of determining information related to planning and scheduling of business activities within an enterprise. This process encompasses all the business activities that create and manage information required for planning and scheduling. These activities are considered from two different points of view: administrative and functional. From the administrative point of view, all the business activities are grouped according to particular business objectives or problem domains such as production management, inventory management or plant engineering. The functional point of view shows functional aspects of business activities, such as plan, order or schedule management functions, which relate these activities to planning and scheduling. Relationships between those two views can be established for every their combination.

In our opinion, such an understanding of APS system is revolutionary ones because it states that APS is not part of ERP but rather, vice versa, ERP has its purpose to support APS system. This fact changes the understanding of what a functional structure of manufacturing enterprise should be and even a philosophy beyond Management Information Systems (MIS) in general. If in the past a MIS first of all was thought of as a transaction processing system, which possibly includes a decision-support subsystem, now it should be thought of first of all as decisions simulation and optimization system supported by a decision-oriented transaction processing system running as one of its subsystems. The main purpose of MIS is to suggest the alternative decisions, which can be made by management, and for each alternative to generate and evaluate scenarios describing possible impacts and consequences. Of course, the system should not pretend to replace the managers. It should only empower the managers by narrowing the set of alternative decisions and allowing them to consider only the near optimal and therefore perspective ones.

Other sources define the term APS system rather at realization level considering them as software products. One of the most popular product-oriented definitions of APS system is given by the Association for Operation Management (APICS). Although APICS dictionary states that APS is a collection of techniques that deal with analysis and planning of logistics and manufacturing during short, intermediate, and long-term time periods, it defines APS system as:

"*... any computer program that uses advanced mathematical algorithms or logic to perform optimization or simulation on finite capacity scheduling, sourcing, capital planning, resource planning, forecasting, demand management, and others. These techniques simultaneously consider a range of constraints and business rules to provide real-time planning and scheduling, decision support, available-to-promise, and capable-to-promise capabilities. APS often generates and evaluates multiple scenarios. Management then selects one scenario to use as the "official plan*" (Blackstone, 2010).

According to product-oriented point of view, APS system does not substitute but only supplement or extend existing ERP systems (Rohde, 2005). The ERP system handles basic activities and transactions, such as customer orders, accounting, etc., whereas the APS system handles daily activities for analysis and decision support. It can be implemented as a subsystem of a particular ERP, as an add-on to a number of different ERP or as a stand-alone package which can be executed autonomous as well as in cooperation with a particular ERP.

Regardless of how an APS system is implemented – as a subsystem of a particular ERP or as an add-on to some number of different ERPs – it is seen as a source of complementary functionality and can be used together with only those ERP systems which are precisely defined in advance (Texeira and Coube, 2009). These two cases, however, differ significantly from the modeling point of view. If an APS system is implemented as a subsystem of a particular ERP, it is an integral part of this ERP and should meet its internal modeling, architectural, and implementation standards. It is not a case if an APS system is implemented as an add-on to ERP and even more so when it is implemented as a standalone software package. In such cases it is not expedient and often even impossible to extend internal ERP standards to the APS system.

Although, in general, APS systems support complex planning and scheduling tasks, it is not simple to explain, what does really mean "*a complex planning and scheduling task*" (Kjellsdotter, 2012). The term APS system is rather ambiguous. Most of the APS systems are industry-specific and differ in many aspects including planning concepts, planning tasks, planning methods, plans' optimization procedures, and even in the used terminology (Kilger and Wetterauer, 2005). On the other hand, they have a number of commonalities and form a family of congenerous systems which is characterized by these commonalities. Grouping can be done at different levels: by common business objectives, by common problem domains, by set of typical problems and by set of typical features. Let us discuss these levels in more details.

At business objective level, the family of APS systems can be defined by the following common objectives (de Santa-Eulalia *et al.*, 2011; Bitran and Tirupati, 1989; Öztürk and Ornek, 2014; Fleischmann *et al.*, 2005): (1) to minimize losses of the material, financial, workforce and other recourses; (2) to better manage the risks and more adequately respond to them; (3) to increase robustness and agility of the production and other plans; (4) to reduce planning time and to improve responsiveness of the planning system; (5) to link the strategic, tactical, and operational level planning decisions in more effective way, emphasizing the integration of planning and scheduling processes; (6) to increase the integration of the internal supply chain; (7) to extend the risk management from an internal to an external supply chain point-of-view (holistic optimization) paying more attention to the stochastic behavior of this chain and to the multi-tier collaboration among its entities (collaborative planning); and (8) to make the order fulfillment and other processes more transparent.

Each APS system supports all or part of the above listed objectives, but even the whole family supports them only to some extent. For example, many APS systems implement algorithms for holistic optimization and collaborative planning, but these problems are indeed not properly solved. The holistic optimization still cannot cope properly with situations where some entities belong to a number of different supply chains. The big challenge remains for the collaborative planning to interconnect different business models because this requires to share strategies, timely information, re-sources, profits and losses among supply chain entities (de Santa-Eulalia *et al.*, 2011). The holistic optimization and collaborative planning problems are properly solved almost exclusively in the cases of centralized control of the whole supply chain. It is because in such cases supply chain management can be seen as a simple extension of an internal integrated supply chain management (Zijm, 2000).

The common problem domains of APS systems were analyzed and defined by a number of authors (de Santa-Eulalia *et al.*, 2011; Meyr *et al.*, 2005; Blackstone, 2010; Stadtler, 2004; Theeuwen, 2007). They are as follows: (1) strategic network planning; (2) demand planning; (3) demand fulfillment and available-to-promise; (4) master planning; (5) production planning and scheduling; (7) distribution planning; (8) transport planning; and (9) purchasing and material requirement planning. Each APS system is able to solve problems in all or in some subset of these domains. Usually it provides a separate subsystem for each problem domain. Many APS systems can solve certain problems in additional problem domain, namely, in: (10) safety stock planning. The purpose of safety stock planning is to install buffers – in the form of either safety stocks or safety times – in order to mitigate impacts of unpredicted business events. However, the safety stock planning cannot be implemented as a separate subsystem because the buffering is a cross-cutting concern. Besides it is an industry-specific procedure depending on the locations of the decoupling points (Tempelmeier, 2001).

Current APS systems include the full spectrum of enterprise and even inter-enterprise planning and scheduling functions (Malindžák *et al.*, 2011). They can plan and schedule work-force, procurements, distribution, sales and other manufacturing related processes. However, APS systems were developed primarily with the aim of improving the production planning and shop floor scheduling. It happened forasmuch the classical ERP systems were rather transactional than planning the. They paid limited attention to the integration of planning and scheduling activities and performed them sequentially, well-nigh ignoring their relationships (Moon and Seo, 2005). A production plan provides the route, processes, process parameters, machines, and tools required for production (Chang and Wysk, 1985). A shop floor schedule allocates the operations to time intervals on the machines. Thus, production planning and shop floor scheduling are highly interrelated and it is impossible to prepare realistic schedules without their proper integration (Maravelias and Sung, 2009; Chen and Ji, 2007). So, at shop floor level classical ERP systems frequently raised serious problems such as varying workloads, changing bottleneck, etc. In some situations, especially in cases when objectives of the production plan and shop floor schedule conflict, no feasible schedule can exist at all (Chen and Ji, 2007). Therefore at shop floor the original plans often were modified (Tan and Khoshnevis, 2000; Chen and Ji, 2007). Mostly this was done in an informal way. The changes, even if they were done following some formal procedure, could not be feed back to the planning level. As a result, discrepancies aroused between plans and production schedules, or solutions overlapped or even partly duplicated each other (Tan and Khoshnevis, 2000). This situation leaded to the integration of production planning and shop floor scheduling activities in APS systems. These systems simultaneously plan and schedule production taking in account available materials, workforce and plant(s) capacities (Bubenik, 2011). Planning and scheduling activities cooperate as partial stages of the integrated process, allowing overlapping the planning of operations with their allocation in time (Garrido and Barber, 2001). The integrated system can simultaneously take into account constraints at both enterprise and plant levels, consider materials and capacity issues together, and integrate production, distribution, and logistics management issues (Hvolby and Steger-Jensen, 2010). As a result, a single optimization problem should be solved. Often this problem is addressed as main (or general) advanced planning and scheduling problem (APSP) (Moon *et al.*, 2004; Moon and Seo, 2005; Zhang and Gen, 2006; Chen and Ji, 2007; Lee *et al.*, 2002).

Although the list of features supported by an APS system depends on branch of industry and many other circumstances, the analysis of the population of such systems shows (Hvolby and Steger-Jensen, 2010; Öztürk and Ornek, 2014; Bubenik, 2011; de Santa-Eulalia *et al.*, 2011) that almost all APS systems have the following features: (1) the planning engine that is based on optimization and constraint-based planning algorithms, is able to generate near optimal plans and use some simulation techniques allowing the simulation different planning scenarios before a plan release; (2) the ability to take into account constraints at enterprise level as well as at plant level what enables to create plans which satisfy multiple objective goals and are near optimal according to financial and other strategic objectives of an enterprise; (3) the ability to consider the potential bottlenecks explicitly; (4) the ability to plan and schedule production taking in account available materials, labor, and plant capacity simultaneously; (5) finite capacity planning at the shop floor level (the sequence in which jobs are carried out effects the set-up time); (6) hierarchical planning encompassing all planning levels from strategic to operational one, and coordinating and integrating vertical and horizontal information flows among problem domains; and (7) integration of forecasting, manufacturing, distribution, and transportation issues. The difference between constraint-based planning and optimization is that constraint-based planning produces feasible but not necessarily optimal plans because only constraints but no plan optimization objectives or criteria are considered. Horizontal flows stream among problems and coordinate the processing of customer-oriented information, such as, for example, customer orders, sales forecasts or purchasing orders (Fleischmann *et al.*, 2005). Vertical flows stream downward and upward among plans of different level and are used to harmonize these plans. For details, the reader is referred to Fleischmann *et al.* (2005).

2.3. *Benefits and Shortcomings of APS*

The potential business benefits of APS systems were discussed by many researchers (e.g., Bubenik, 2011; Texeira and Coube, 2009; Kjellsdotter, 2012; de Santa-Eulalia *et al.*, 2011; Öztürk and Ornek, 2014; Fleischmann *et al.*, 2005; Kjellsdotter, 2009; Chen *et al.*, 2012; Kjellsdotter and Jonsson, 2010; Kjellsdotter, 2012 and Stadtler and Kilger, 2005; Wortmann, 1998; Kennerley and Neely, 2001) and reported by a number of industrial bodies (e.g., EyeOn, 2014; Blackstone, 2010; Aziz, 2002; Kallrath and Maind, 2006; Consortium PSLX, 2005; HInQu Informatics, 2014). These benefits can be summarized shortly in as follows. APS systems significantly improve the quality of plans due to the simultaneous consideration of both business rules and range of constraints at enterprise as well as at plant level and due the possibility to evaluate multiple alternative "what-if" scenarios. Production plans prepared by such systems are harmonized, feasible with realistic manufacturing deadlines. They integrate manufacturing, distribution, and transportation issues. A single plan provides both long-term aggregate activities and short-term operations performed at shop floor. Multiple planners can access data from this plan simultaneously. The potential of personnel, machines, material and other resources is used in more performance-supporting and cost-effective way, materials and capacity issues are considered simultaneously. APS systems allow multi-site, multi-product planning, scheduling and control in real-time, taking into account the dynamic nature of production process. Due to the prompt planning and near-optimal scheduling, APS systems significantly reduce orders' throughput times. They can plan all supply chain facilities simultaneously and helps to synchronize hundreds of planning decisions at strategic, tactical and operational levels across the whole supply chain. APS systems allow creating plans satisfying the multiple objective goals. Potential bottlenecks are considered explicitly. Trading partners can be effectively involved in the supply chain planning process because the schedule information can be shared, workflow-driven exception messages about the critical issues across the whole supply chain can be sent simultaneously to both internal planners and trading partners and the required feedback can be received from these partners immediately. Finally, APS systems work with an implementation-independent abstract data model and therefore are insensible to the changes in accompanying software caused by the modification of business processes.

On the other hand, APS systems are not a silver bullet. They have also a number of shortcomings. The optimism about the real possibility to achieve potential business benefits promised by APS systems flagged already about ten years ago after a number of ambitious APS-based projects failed to do this Kilger (2005). A significant limitation of APS systems is their ability to cope with the uncertainties (Ptak and Smith, 2008) Despite the robust planning (van Landeghem and Vanmaele, 2002; Aghezzaf *et al.*, 2011; Pimentel and Brem, 1994; Aghezzaf, 2010) and the other theoretical achievements (Kim *et al.*, 2011; Al-E-Hashem *et al.*, 2011), this still remains a serious problem (Wu *et al.*, 2011). For exhaustive discussion on this topic the reader is referred to Graves (2011).

A serious shortcoming of APS systems is their need for tremendous amount of data required for the optimization of plans and schedules. The scheduling typically requires detailed routing information about resource requirements (operation sequence, primary and secondary resources, machine-specific processing times, sequence-specific setup time, parallel operations, alternate operations, material requirements tied to an operation) (Hamilton, 2014). In addition, the availability of resources permanently changes and data about these changes (machine down time, hourly changes in head counts, etc.) also are required. It is far not simple to maintain such amount of data.

Another significant shortcoming of current APS systems is their inability to support properly negotiation and collaboration schemas among trade partners (de Santa-Eulalia *et al.*, 2011). The negotiation abilities are vital, because trade partners maintain different business models and pursue different business goals. Although in theory, a single APS system is able to optimize all plans across the whole supply chain, in practice this is usually impossible because of different, sometimes even contradictory objectives of different partners. In addition, many enterprises still cannot achieve the integration of their internal supply chains. Thus, according to de Santa-Eulalia *et al.* (2011), current "APS systems faces important barriers related to interconnection among business models, which requires sharing strategies, timely information, resources, profits and loss".

In general, the ability to optimize only a production plan across the trade partners in the supply chain still remains a great challenge (Hvolby and Steger-Jensen, 2010). Although some industrial APS systems are able to do this, this ability is used rare. The stumbling-block for this that each trade partner usually belongs to a number of different supply chains. To quote Hvolby and Steger-Jensen, "Each partner needs a plan which covers all their operations and not a plan which only optimizes a subset of their operations" (Hvolby and Steger-Jensen, 2010).

The serious difficulty for current APS system is the simulation in sophisticated supply chain environment (de Santa-Eulalia *et al.*, 2011). As an attempt to overcome this drawback, so-called distributed APS systems were proposed. Such systems use for simulation the agent-based modeling approach, which "models the supply chain as a set of semi-autonomous and collaborative entities acting together to coordinate their decentralized plans" (de Santa-Eulalia *et al.*, 2011). However, the distributed APS systems still are not enough mature and should be seen rather as experimental ones.

There is also a shortcoming of APS systems of other nature. It is similar to troubles with old-fashioned expert systems when users were not able to grasp reasons of conclusions generated by a system and an explanation of chain of reasoning was required. APS systems generate schedules automatically. Finite scheduling rules used for this aim usually are very complex and difficult to understand (Hamilton, 2014). However, current APS system present no explanation on what reasons is based the suggested schedule and managers mistrust them.

The shortcoming of APS systems is also the existence of a number of gaps between the theory of these systems and the industrial practices. For the discussion about these gaps the reader is referred to Lin *et al.* (2007).

Finally, the current APS systems is still too expensive to SMEs due to high acquisition and maintenance costs (Texeira and Coube, 2009).

In summary, "*the optimization of entire supply chains still remains an elusive problem, especially if the objective is to integrate planning with scheduling in these supply chains*" (Grossmann, 2012).

3. Modeling and Implementation Challenges

Despite the relatively long APS systems history, the theory of these systems is rather in its rudimentary phase. There exist a number of APS systems modeling and implementation problems which should be investigated in more detail, theoretically sound solutions of these problems should be designed and incorporated into the main theory of information systems engineering. Bellow we discuss the most important APS systems modeling and implementation challenges identified partly as a result of the study of scientific literature, partly through an in-depth analysis of the experience gained during the development of the PEN system.

***Challenge* 1:** Probably, most important APS system modeling challenge is the decomposition of APSP into sub-problems, selection sub-problems' solution algorithms and forming them into a cohesive ensemble that solves the whole APSP. This problem is also the most fundamental one. To solve it, the question "How to design the possibly best ensemble of collaborating algorithms required for a particular APS system?" must be answered. The Chapter 4 examines this problem in detail and explains the meaning of the term "an ensemble of collaborating algorithms".

***Challenge* 2:** Another important APS system modeling challenge closely related to the first one is the division of labor among optimization, simulation and, possibly, other approaches. The question "How optimization, simulation and, possibly, other planning methods should be combined together in a particular APS system?" must be answered in order to cope with this challenge.

***Challenge* 3:** Finally, the last identified important modeling challenge is the combination of different kinds of models in a particular APS system. In other words, the question "How to combine descriptive forecasting models (e.g. for forecasting future demand or manufacturing costs), simulation models, optimization models, models modeling uncertainty and abstract data models?" must be answered.

***Challenge* 4:** The integration of APS and ERP systems is one of the most important implementation challenges. In order to cope with this challenge, a number of complex research questions must be answered: "What are the valid patterns of APS and ERP systems collaboration, taking into account that ERP systems are MRP-based systems and APS systems use finite capacity scheduling approach and optimization?", "What an architectural solution can be used in a supply chain environment for integration of a particular APS system with the variety of partners' ERP systems?", "What the collaboration and architectural patterns can be used to integrate APS system developed as an add-in to different ERP systems the list of which is unknown in advance?", "How data should be transferred between ERP and APS systems?", and "How the labor should be divided among APS and ERP systems?"

***Challenge* 5:** The reference modeling of a particular APS system is another important implementation challenge. It entails a number of complex research questions: "Should a reference model for a particular APS system be hierarchical one or not?", "What conceptual basis should be used to develop this model?", "To which extent the reference model should be platform-oriented?", "Should it define also concepts which are required to describe various algorithms implemented in this APS system?", "How to combine "best practice" reference models with "system oriented" reference models and how communicate clearly ideas and knowledge captured in the reference model to two such different communities, namely, decision makers and engineers?", "In which way a reference model for a particular variable APS system should define the concepts that knowledge captured in this model would be easy transformable into a specific model of a particular target enterprise?", and "Which notations are most appropriate to describe different issues in a Reference Model document and how to define the semantic of concepts used in this document?"

4. Integration of APSP Decomposition, Optimization and Simulation Algorithms

4.1. *An Ensemble of Collaborating Algorithms*

It became possible to implement APS systems only when the memory resident servers have been developed. Such servers allow storing the entire supply chain planning engine and its environment including optimization and simulation models, and all required data in the internal memory. However, despite this and tremendous advances in the development of hardware and sophisticated software, it is still impossible to solve APSP as a monolithic optimization. This is impractical even in case when we aim to find only near-optimal solution.

One of the simplest formulations of APSP is given by Wei Tan and Behrokh Khosnevis:

"*…given a set of n features which are to be processed on m machines with alternative process plans and other technological constraints for each feature, find a process plan for each feature and a sequence in which features pass between machines and a sequence in which features on the same part are processed such that it satisfies the technological constraints and it is optimal with respect to some performance criterion*" (Tan and Khoshnevis, 2000).

This formulation is very abstract because it does not explain what concrete "technological constraints" must be satisfied. The simplest technological constrains are precedence and available capacity constraints. Precedence constraints define which operation sequences are technically feasible. Available capacity constraints describe the capacity of each resource in a given time period. Note that available capacities vary over time. For example the capacity of the whole plant in any period t depends on the operational states of machines. It changes as machines fail or are under repair. If we assume that all available capacities are stable and the only aim is to complete all orders in time, the global optimization of all plans and schedules across the whole supply chain is still practically impossible. The simplest (classical) scheduling problem for a one-plant enterprise is formulated as follows: "*…the actual assignment of starting and/or completion dates to operations or groups of operations to show when these must be done if the* *manufacturing order is to be completed on time*" (Blackstone, 2010). This problem is constrained by (a) precedence constraints (sequencing) and (b) stable available capacity constraints (deterministic processing times of all machines, no setups, no machine breakdowns, no pre-emption and no cancellation). It is required to find the optimal technically feasible sequences of operations that meet capacity constraints. However, even this problem is NP-hard. Real-world production scheduling problems are much more complicated. For exhaustive discussion on this topic the reader is referred to (Herrmann, 2006; de Carvalho and Haddad, 2012).

The above given APSP formulation states that the scheduling is only one among many sub-problems of APSP. Therefore, the whole APSP is yet more complicated than any of its sub-problem. However, in this formulation neither the network of production plants of an enterprise nor the more a multi-site nature of a whole supply chain is taking into account. Generally, the APSP can be very sophisticated. It can be formulated at different abstraction levels taking into account a number of various constraints at enterprise level as well as at plant level and considering a number of various optimization objectives. The most popular objective is to minimize the makespan, i.e. total length of a schedule (Zijm, 2000). However, a number of other objectives, for example, the maximization of equipment utilization or the maximization of the probability of meeting the due date, is also often used. Besides, APSP may be formulated for the multi-plant enterprises or even for a whole multi-site supply chain. In short, we should consider not a one well-defined planning and scheduling problem but a big family of quietly different production planning and scheduling problems, including very complicated ones. For example, in a multi-plant enterprise the manufacturing system is composed by a network of hierarchically integrated production plants. Every plant has its own resources with different functions, processing times and capabilities. However, for the reason that it is a part of network, its relation with other plants is as important as its internal model (Alvarez, 2007; Gnonia *et al.*, 2003). Normally, the output from one plant becomes an input into another plant. In multi-plant manufacturing system, it is necessary to consider also the outsourcing (Lee *et al.*, 2002), i.e. situations when "*the same order can also be delivered to some other plant for assigning to the resources in different locations*" (Zhang and Gen, 2006). In other words, we have some sequential and parallel machining processes structure that can be represented by an AND/OR directed graph (Homem de Mello and Sanderson, 1990). For detail discussion on the integrated planning and scheduling in multi-plant environment the reader is referred to Kanyalkar and Kadil (2005).

From production planning perspective, an industrial multi-plant enterprise is a multi-level manufacturing system, at each level of which local performance objectives are defined (Behdani *et al.*, 2010). These objectives are hierarchically interconnected and should optimally contribute to the overall goal. Due the rather fuzzy relations between the local and overall performance goals, the long time horizon and high abstraction level of the overall goal, the APSP significantly grows in complexity. The main question that should be answered is: "*How to best operate a multi-plant enterprise and the separate plants at the same time*?" (Behdani *et al.*, 2010).

Using representation of multi-plant manufacturing system by an AND/OR directed graph, APSP for such enterprise can be formulated as a traveling salesman problem with precedence constraints (Moon *et al.*, 2002; Oboulhas *et al.*, 2005; Sung and Jeong, 2014). The classical stand-alone traveling salesman problem is already NP-hard. It becomes even more complicated with precedence or other additional constraints. Hence, in many cases the overall APSP is very difficult, if it is considered as monolithic, and can be solved only by some evolutionary algorithm or through hybrid approach that combines several algorithms of a different nature.

In more sophisticated manufacturing environments (e.g. multi-product system, alternative machines, flexible operations' sequences, etc), under the realistic assumptions, constrained by some additional constraints (e.g. multi-level product structure, multi-echelon production, etc.) and with the objective function containing earliness and tardiness penalties the APSP is even more complex. Besides, the scheduling problem depends on shop type (flow shop, job shop, hybrid shop, etc.) (Tan and Khoshnevis, 2000).

To sum up, the integrated planning and scheduling problem cannot be solved as a monolithic optimization problem (Phanden *et al.*, 2013). Exact mathematical optimization methods have been designed to find optimal solutions satisfying the given technical-economic objectives. However they are time consuming and cannot solve real-world planning and scheduling problems in a reasonable time. Due to this reason, in most APS systems the mathematical optimization methods should be combined with some heuristic optimization methods, including simulation and imitation based ones, which, according to Cote and Laughton (1984), "have performed surprisingly well in their search for near-optimal solutions".

Simulation and imitation are kinds of heuristic methods. The necessity for simulation methods arises from the fact that mathematical optimization models are based on the restrictive assumption about the static nature of manufacturing systems. However, neither the dynamic characteristics of real-world manufacturing systems nor the stochastic nature of abnormal situations such as machine failure can be properly modeled by such models. Real-world manufacturing systems operate in a dynamic environment under uncertainties. For this reason, optimal and near-optimal solutions searched by optimization methods are often inadequate. On the other hand, the simulation models explicitly models inside (endogenous) as well as outside (exogenous) manufacturing system aspects, but they are inadequate be used to find optimal solutions (Safaei *et al.*, 2010; Gnonia *et al.*, 2003). Simulation methods are based on heuristics and, consequently, cannot precisely implement complex decisions logic required to choose the best solution. So, in general case, exact optimization and heuristic methods including simulation and imitation ones should collaborate solving APSP.

In summary, APSP should be decomposed into a complex of sub-problems of manageable sizes and *an ensemble of collaborating algorithms* should be designed to solve these sub-problems. By *an ensemble of collaborating algorithms* we mean a problem-oriented arrangement of a collection of exact or/and heuristic optimization, constraint programming, simulation, and imitation algorithms, which:

- implements some *collaboration pattern* or a composition of such patterns;
- is designed taking into account the computational resource constraints and the specific features of some subclass of APSP problems;

- optimizes the overall performance of APS system under consideration; and
- seeks to find at least near optimal solution of APSP problem in a reasonable time.

In an ensemble, information is passed back and forth among the subproblems until an acceptable solution to the whole APSP is achieved. The idea of an ensemble of collaborating algorithms is far not new. Its origins are in the seminal work of Minslry (1988), where he in fact proposed to construct problem solving agents as compositions of interacting, simpler subagents. This idea has been explored and developed further in a variety of different fields by many authors, but, to the best our knowledge, was never formulated explicitly in the context of APS systems.

The APSP can be decomposed into sub-problems in a number of different ways. By a *collaboration* pattern we mean a kind of behavioral patterns, which describes an abstract arrangement of solution algorithms for APSP sub-problems and a way they interact each with other for some kind of APSP decomposition. Thus we have several categories (hierarchical, iterative, etc.) of collaboration patterns. Any category describes the roles, which corresponding algorithms play in the APSP solution process. Roles are allocated to sub-problems. A role is specified by its goal (i.e. output requirements) and a solution method (e.g., MILP or event-based simulation) of the sub-problem to which it is allocated. Algorithms are assigned to roles during the design of an ensemble of collaborating algorithms for a particular APS system. We call this problem a *role assignment problem*.

To the best our knowledge, the term "*collaboration pattern*" currently is used neither in the context of APS systems nor in the context of mathematical programming theory. Research papers on mathematical programming or on production planning and scheduling usually address such the patterns as "*problem decomposition methods*". Several other terms with similar meaning are also in use. For example, in constraint satisfaction field the term "*algorithmic chaining*" (Borrett and Tsang, 2009) is used to address the switching from one algorithm to another during the problem solving process. Although algorithmic chaining also deals with the arrangements of algorithms, these algorithms are arranged in a pre-determined order. Besides, it is used for quite different purposes, namely, to improve the overall performance of *algorithms portfolio* application (Huberman *et al.*, 1993). Note that an algorithm portfolio is a set of algorithms that are bundled together to increase overall performance and/or the quality of final results. The algorithmic chaining performs in the following way. It starts the first algorithm from the arrangement. The switching mechanism, which is a part of the algorithmic chaining system, monitors the execution of this algorithm and stops it when certain conditions occur. After this, the algorithmic chaining starts the next algorithm. The same is repeated for each algorithm (Borrett and Tsang, 2009). Shortly, the purpose of the algorithmic chaining is not the same as the purpose of a collaboration pattern. Not about the collaboration of algorithms is also the concept of hybrid algorithm, which is defined as "*a collection of heuristics, paired with a polynomial time selector S that runs on the input to decide which heuristic should be executed to solve the problem. Hybrid algorithms are interesting in scenarios where the selector must decide between heuristics that are "good" with respect to different complexity measures*" (Vassilevska *et al.*, 2006). However, both these concepts – algorithmic chaining and hybrid algorithm – as well as the concept of algorithms portfolio are useful and applicable in solving the role assignment problem.

Finally, the term "*ensemble*" can also be found in the context of algorithms portfolio (Dietterich, 2000). According to this source, "*An ensemble of classifiers is a set of classifiers whose individual decisions are combined in some way (typically by weighted or un-weighted voting) to classify new examples. ...ensembles are often much more accurate than the individual classifiers that make them up*". Thus, in Dietterich (2000) this term addresses an instance of algorithms portfolio (Kottho, 2012), which is designed to improve the accuracy of results. So, the term "*ensemble*", as is used in Dietterich (2000), has also the other meaning than in our case. However, it can also be useful at the algorithms assigning phase because an ensemble of collaborating algorithms should be designed keeping in mind both the performance and the accuracy evaluated applying near-optimality criteria.

The design of the best or, at least acceptable ensemble of collaborating algorithms required for a particular APS system is a serious implementation challenge. No standard solution exists for this challenge. The nature of APSP depends on the specific of a particular industry, particular shop type, kind of integrated production planning and scheduling problem and other factors. In terms coined by John R. Rice in his famous paper "The Algorithm Selection Problem" (Rice, 1975), each particular APS system has its own problem space, where the term "*problem space*" refers to a class of individual problems or instances having the same form (Tovey, 2002). In our case, the problem space of particular APS system is a subclass of the class of all APSP.

In the context of algorithm selection problem (ASP), the term "algorithm" can refer to a system, a program, a heuristic, a classifier or a configuration (Kottho, 2012). An ensemble of collaborating algorithms, like as software system, is a composition of algorithms and in the context of ASP can also be addressed as an "algorithm". Thus the role assignment problem can be seen as a specific algorithm selection problem, in which the algorithm space consists of several algorithm portfolios (optimization algorithms, simulation algorithms, etc.). Portfolios are related to the roles and at least one algorithm should be selected from each portfolio. All these algorithms should be composed together to form an effective working ensemble of collaborating algorithms, which hopefully computes near-optimal solution to the APSP. Note that in this context, the role assignment problem should be considered as a functionality allocation problem, which, generally, is less understood than a resource allocation problem. Given that, in general case, a decomposition of APSP can be multilevel (i.e. any sub-problem can be decomposed further) and that collaboration patterns of different categories can be required at different levels. So, the collaboration patterns for a particular APS system can be of hybrid nature and very complex. Consequently, the role assignment problem often can be very hard. Moreover, the algorithms of different nature are far not easily composable into the ensembles. Currently there exists no systematic procedure to solve this problem. It is usually solved in an ad hoc manner. Anyway, it is possible to compare, at least informally, the different alternatives in functionality allocation against the chosen criteria. So, the question arises, what the criteria and measures should be used to evaluate a quality of an ensemble of collaborating algorithms.

On the basis of reviewed research works and our practical experience, we conclude that the quality of an ensemble of collaborating algorithms should be evaluated according to at least the following four criteria:

***Criterion* 1:** Efficiency criterion: how high is the computational cost of the selected ensemble of collaborating algorithms? A theoretical measure to evaluate the efficiency of an algorithm is its complexity in asymptotic sense. This measure can be generalized to apply it also to the ensembles of collaborating algorithms. However, the complexity measure is of little usefulness for practical evaluation of computational cost of an ensemble. It is so because usually, when evaluating an ensemble, it is important not to evaluate the behavior of a particular ensemble for the whole problem space but only for some subset of its realistic instances (Tovey, 2002). Besides, such ensemble almost always includes algorithms, especially heuristic ones, for which computational cost varies greatly from one instance of the problem space of given APS system to another (Huberman *et al.*, 1993). According to Huberman *et al.* (1993), such unpredictable variation in computational cost can be characterized by a distribution describing the probability of obtaining each possible computational cost value. Therefore, the mean or expected values of these distributions can be used as an efficiency measure of an ensemble of collaborating algorithms.

***Criterion* 2:** Finiteness criterion: what is the risk that the selected ensemble of collaborating algorithms for some instances of the APS system problem space will not produce an acceptable solution in a reasonable time or even at all? This risk arises even in case of some exact optimization algorithms. In case of simulation and other heuristic algorithms, it exists almost always. According to Huberman *et al.* (1993), this risk can be characterized by the standard deviation of the distribution mentioned in the Criterion 1. We remind that the standard deviation describes how likely it is that for particular instances of the problem the computational cost will deviates from the expected one.

***Criterion* 3:** Correctness criterion: how close to optimal are final feasible solutions computed by APS system? An approximation ratio measure can be used to evaluate the selected ensemble according to this criterion. For example, the average case or worst case approximation ratio or both measures can be used. Approximation ratio can be absolute or relative. The behavior of algorithms on instances with relatively large optimal value can be evaluated by an asymptotic approximation ratio measure. For the definition of the above mentioned measures the readers are addressed to Section 8 in Vidar (2005) and to Epstein and van Stee (2008).

***Criterion* 4:** Robustness criterion: in what degree the plans and schedules prepared by an ensemble of collaborating algorithms are "able to undergo perturbations without being invalidated" (Hebrard, 2007)? There are many definitions of the robustness. Generally, it can be considered as an umbrella-concept for the whole family of related concepts, such as *reliability, stability, fault tolerance, adaptability, correctness, uncertainty processing, risk-sensitiveness*, etc. In the distributed environment it can be even understood as a *safety*. Sometimes the robustness means the ability of an algorithm to continue working despite abnormalities in input, calculations, etc.; sometimes the ability to produce the satisfactory result when the constant parameters in the formulation of the problem are perturbed (Chiang *et al.*, 2007) or even the model of the problem is slightly changed. Robustness becomes especially important when the problem is transient or unstable (Tovey, 2002). It is also noteworthy that the trade-off between robustness and performance exists. The most important approaches for evaluation of robustness are the sensitive analysis (Castillo *et al.*, 2008), stochastic programming (Esmaeili *et al.*, 2013), robust optimization (Mulvey *et al.*, 1995; Bertsimas *et al.*, 2011), and fuzzy programming (Vidar, 2004). In our opinion, the most promising approach to evaluate robustness of an ensemble of collaborating algorithms is described in Gupta *et al.* (2006).

***Criterion* 5:** Implementability criterion: how easy it is to compose the selected algorithms and implement the obtained composition as an ensemble of collaborating algorithms? The measures to evaluate an ensemble of collaborating algorithms according to this criterion are time and amount of money. These measures, in turn, depend on the competence of an implementation team, available implementation platform and many other factors.

In addition, the quality of an ensemble of collaborating algorithms depends on the proper selection of collaboration pattern because namely it determines the contribution of each algorithm to the whole APSP solution process. In the context of APS systems, the main idea of any collaboration pattern can be described as follows:

- APSP is decomposed into a number of sub-problems of lower dimension;
- each sub-problem is solved separately;
- a solution to the whole APSP is produced on the basis of obtained solutions by using iteration, composition or some other process.

There are many kinds of collaboration patterns including hierarchical, incremental, iterative, and recursive ones. The discussion on classification of collaboration patterns and investigation of their properties is out of the scope of our paper. It is the area for special research. Here, as illustrative examples, we discuss shortly only hierarchical, incremental, and iterative patterns.

4.2. *Hierarchical Collaboration Pattern*

A hierarchical collaboration pattern is one of the most popular. It is used to implement various hierarchical production planning methods (Bitran and Tirupati, 1989; Saad, 1990; Baumann and Dimitrov, 2008; Neureuther, 2004; Hax and Candea, 1984; Hax and Meal, 1975). This pattern provides an interaction scheme between APSP sub-problems that is analogous to the interaction scheme between activities in the well-known systems development life cycle model (SDLC) – waterfall model (Royce, 1970; Bassil, 2012). The planning and scheduling approaches, which are based on the hierarchical collaboration pattern, often are referred to as the waterfall-style methods (e.g. in Myers *et al.*, 2001) or, especially in the literature on decision making, as a rational comprehensive planning model (e.g. in Cabantous and Gond, 2011). The philosophical roots of this approach are in logical positivism.

A hierarchical collaboration pattern usually is used to decompose the APSP into three sub-problems: strategic or long term planning, tactical or midterm planning, and operational or short term planning. However, sometimes the ASPS is decomposed only into two – planning and scheduling – sub-problems or into more than three sub-problems. In any case, the whole APSP solution process proceeds in a top-down manner. Solutions to higher level sub-problems impose the constraints to satisfy and/or the objectives to attain on lower level sub-problems. In turn, the solutions to lower level sub-problems are considered as a feedback to evaluate the quality of higher level solutions. If the compatibility between the higher and lower levels solutions is violated, an emergence procedure, which changes higher level solution, should be triggered (Dauzere-Peres and Lasserre, 2003).

The plans, which are produced by solving sub-problems of different level, differ in the scope of the planning activity, planning horizon, level of detail, and the authority and responsibility of the manager in charge of executing the plan (Bitran and Tirupati, 1989). At the upper level we have long time strategic plans, at the bottom level – detailed schedules. The higher and lower level plans are related by the aggregation/disaggregation relation-ships (Axsater and Jonsson, 1984). So, the higher level plans are "*disaggregated in time and detailed by products and items, taking into account some basic constraints originating from operational levels*" (Hennet, 1999). To solve each sub-problem is simpler than the whole ASPS because the 'details' that are not relevant to the level of this sub-problem are ignored. In traditional hierarchical approach, the upper level sub-problems are typically modeled as linear or mixed integer-linear programming (MIP) problems (Chen and Ji, 2007; Kanyalkar and Kadil, 2005) and the bottom level sub-problems as convex rucksack problems (Bitran and Hax, 1981). However, the hierarchical collaboration pattern is quite general. It can be implemented using a number of different aggregation and disaggregation schemes and modeling the sub-problems by various optimization models. Its implementations may also differ in how the different level sub-problems (optimization models) interact each with other. The basic shortcoming of hierarchical collaboration pattern is its post-positivist point of view, mainly, the focus on predictability. It is supposed that the manufacturing process is basically predictable and the plans are self-contained. It means that it is possible to predict everything already at the top level of hierarchy and to prepare such plans which can be treated as formal specifications that should be scrupulous implemented by lower level plans and schedules. Indeed, it is impossible to produce such schedules, which are "***consistent** with the aggregate plan solution, **feasible** for implementation and **optimal** in minimizing costs and backloggings*" (Saad, 1990). Plans are "*context-dependent, dynamic entities, which are affected by moment-to-moment changes in the environment*" (Maravelias and Sung, 1994) and carefully prepared higher level plans quickly become obsolete. Generally, the pattern suffers from the sub-optimal or infeasible solutions (Kanyalkar and Kadil, 2005). It is mostly because of the unsound feedback process from lower level sub-problems back to higher level sub-problems (Aardal and Larsson, 1990). To the best our knowledge, a feedback process, which is based on sound mathematical decomposition principle and is able to guarantee the feasibility and the optimality of the overall production plan, is still under development. Nevertheless, in some APS systems, especially in SME-oriented ones, such patterns may be successfully applied (Saad, 1990; Baumann and Dimitrov, 2008). Besides there exist a rich body of literature considering how to apply hierarchical collaboration pattern decomposing APSP in various more sophisticated ways. For example, in Quadt and Kuhn (2005) the APSP is decomposed along the production stages. This decomposition is used to solve an integrated lot-sizing and scheduling problem for flexible flow lines. It decomposes APSP into a bottleneck planning, schedule roll-out and product-to-slot assignment subproblems. A bottleneck planning subproblem is being solved separately from the other subproblems, which afterwards both are solved simultaneously. However, when preparing a schedule for the bottleneck stage, the other production stages are implicitly taken into account. Bottleneck planning subproblem is solved using a heuristic procedure, which is based on a mixed integer programming model. This model models explicitly only part of problem elements and approximates the other elements. The schedule roll-out subproblem is constrained by the number of machines per product family and the batch-sizes predetermined by the bottleneck planning subproblem. A schedule roll-out subproblem plans the production at the product families level and as-signs a production time and a machine to each product family unit (Quadt and Kuhn, 2005). Finally, the product-to-slot assignment subproblem disaggregates the product families and determines "*how many and which machines to set up and when to produce the individual products*" (Quadt and Kuhn, 2005). It is constrained by the machine/time slots predetermined by the schedule roll-out subproblem. Due to these constraints, the subproblem can be solved separately for each product family. It is decomposed further into four lower level sub-problems, which determine respectively "(1) how many machines to set up for each specific product, (2) in which batch size, (3) when in the given time window and (4) on which of the predetermined machines to produce the product units" (Quadt and Kuhn, 2005). These sub-problems are solved using nested genetic algorithms.

This example illustrates that the structure of a particular hierarchical collaboration pattern can be very complex and that, in general case, the design of such a pattern can be a great challenge.

4.3. *Incremental Collaboration Pattern*

The philosophy behind this pattern is analogous to the philosophy behind SDLC incremental model (Larman and Basil, 2003; Olsen, 2006; Northover *et al.*, 2008; Lindblom, 1959). The roots of incrementalism, in turn, are in the epistemology of pragmatism (Halton, 2004). In contrast with deduction based positivism, pragmatism uses abductive reasoning. The incremental approach is employed in various application domains and is referred to by a number of different terms (Ansari *et al.*, 2013), for example: "*continuous improvement*" (Nicholas, 2011), "*piecemeal engineering*" (Popper, 2003), "*bounded rationality*" (Simon, 2007), "*muddling through*" (Lindblom, 1959), "*disjointed incrementalism*" (Braybrooke and Lindblom, 1963), "*logical incrementalism*" (Quinn, 1980), "*Kaizen*" (Lolidis, 2008).

The main idea of the incremental approach is vertical separation of concerns (SoC) (Dijkstra, 1982). A concern is a matter of interest (Glinz, 2007). In Adda *et al.* (2010), SoC is defined as "*a general problem solving heuristic that consists of solving a problem by addressing its constraints, first separately, and then combining the partial solutions with the expectation that, (1) they be composable, and (2) the resulting solution is nearly optimal*". So, it is a technique for managing the complexity by decomposing because it breaks the complexity of a problem into loosely-coupled, easier to solve, sub-problems with the hope that their solutions can be relatively easy composed into the solution to the original problem (Adda *et al.*, 2010). Problems can be decomposed either horizontally or vertically. An example of a horizontal decomposition is a hierarchical one which decomposes the problem into a number of horizontal abstraction levels. A vertical decomposition splits the problem into a number of loosely coupled vertical slices that crosscut horizontal abstraction levels. Vertical slices correspond to the sub-problems which, in ideal case, should be solvable independently, without concern about other sub-problems.

There are several kinds of vertical decomposition. A simplest one – we call it proper vertical decomposition – is based on the separation concept in the sense as in Geoffrion and Marsten (1972).[2] According to this work, a set of sub-problems $P_1, P_2, \dots, P_n$ of an optimization problem P with a set of feasible solutions $F(P)$ is the proper vertical decomposition of this problem if the following conditions hold:

1. $\forall x((x \in F(P)) \rightarrow \exists x_i((x_i \in F(P_i))\&(x_i = x))\&\forall j(((x = x_j)\&(x_j \in F(P_j)) \rightarrow (i = j))))$,

2. $\forall i((x_i \in F(P_i)) \rightarrow \exists x((x \in F(P))\&(x_i \in x)\&\forall j((x_j \in F(P_j))\&(x_j = x)) \rightarrow (i = j)))$,

where $1 \leqslant i \leqslant n$, $1 \leqslant j \leqslant n$. In other words, the sequential decomposition defines the partition $F(P_1), F(P_2), \dots, F(P_n)$ on the set $F(P)$. The optimal solution to problem P is the best solution found to any sub-problem. Unfortunately, the proper vertical decomposition of the APSP is possible only when its model has a special structure. For example, it can be done if APSP can be formulated as an integer programming problem with contradictory constraints defined for some integer variable (Geoffrion and Marsten, 1972) or as a nonlinear convex optimization problem for which both objective function and all constraint functions are additively separable ones. An additively separable function is a function which can be expressed as a sum of single variable functions. Therefore, the proper vertical decomposition approximates the original problem by a linear program.

More elaborated kind of vertical decomposition is partially ordered decomposition described in Karimian and Herrmann (2009).[3] Applying this decomposition to the problem P, we obtain a partially ordered set of sub-problems, in which "*[t]he solution to one subproblem will provide the inputs to one or more subsequent subproblems*" (Karimian and Herrmann, 2009), that is, the solution to any sub-problem, except the last one, is used to solve next sub-problem. It means that, solving a sub-problem, decisions, which already were made solving previous sub-problems, are taken into account. However, the ordering of the sub-problems does not necessarily reflect the importance of their objectives. Nevertheless, the solution is constructed incrementally.

The objective function F of the problem P in any sub-problem P_i should be replaced by a surrogate objective function F_i. Surrogate functions are obtained, removing decision variables that are not relevant to this sub-problem and will be defined later in other sub-problems, and making other application-oriented modifications. The constraints in any

[2]Currently, in the literature on mathematical programming the term separable programming is mostly used.

[3]Authors use the term *separation* and reserve the term *decomposition* for the cases when sub-problems are coordinated by a special second-level sub-problem usually referred to as a *master problem*.

sub-problem also should be modified appropriately. It means that the vertical partially ordered decomposition is highly informal. Defining surrogate objectives and appropriate constraints may require informal domain knowledge "*about which issues are the most important ones and which solutions are usually poor ones*" (Karimian and Herrmann, 2009). This is a disadvantage of this approach. Another disadvantage is that such decomposition not in all cases produces an optimal solution to the original problem P. On the other hand, due the partial ordering of sub-problems, some sub-problems can be solved in parallel. This is an advantage of this approach.

It seems that in APS systems the vertical partially ordered decomposition currently rather is not used. At least, we did not find any work on this topic. Nevertheless, this approach is interesting as an example of the simplest incremental decomposition.

The most challenging problems arise in the vertical separation of concerns when global and/or crosscutting concerns are present. Global concerns are concerns that affect the whole problem P or its significant part (Aldrich, 2000). Examples of global concerns are complicating variables and complicating constraints. Complicating variables are those which prevent a straightforward solution of the APSP by blocks (Conejo *et al.*, 2006). For example, in the context of APSP with integer and continuous variables, integer variables usually are considered as the complicating ones because treating integer variables is much more complicated than treating continuous variables. Complicating constraints are those that involve variables from different blocks of the constraints matrix and, like complicating variables, also prevent a straightforward solution of the APSP by blocks (Conejo *et al.*, 2006). Crosscutting concerns are those that cross cut functional concerns of the problem P. Mainly, they are some algorithmic properties. An example of crosscutting APSP concern is robustness of algorithms. Both kinds of concerns prevent the decomposition of the problem P into completely independent sub-problems implementing different functional concerns but should be handled quietly different. Crosscutting concerns should be either handled in a centralized manner or scattered throughout functional concerns. In the context of software engineering, usually so-called aspect-oriented (AO) techniques (Solberg *et al.*, 2005) are used for this aim. There exist a number of such techniques, including weaving-based, interception-based approaches, dependency injection, etc. Using AO techniques, crosscutting concerns are automatically weaved with separable functional concerns, injected into these concerns or processed in each functional concern after or before it intercepts some event. However, these approaches are rather implementation-oriented, based on specific linguistic and meta-linguistic mechanisms. The body of literature considering how to adapt these techniques for the algorithmic level and apply designing an ensemble of collaborating algorithms currently is miserable.

There are several approaches how to handle global concerns. All these approaches are based on the relaxation concept (Geoffrion and Marsten, 1972) and decompose the original problem P into smaller sub-problems which are coordinated by so-called master problem. Often it is referred as a bi-level decomposition.[4] The convergence towards the

[4]Many authors (e.g., Conejo *et al.*, 2006) use the term "*bi-level decomposition*" in more narrow sense, namely, to address the case when evaluation of a constraint function requires to solve its own optimization problem. So, the original problem is decomposed into two optimization problems. "*Nevertheless, the structure of the problem suggests a Benders type decomposition mechanism*" (Conejo *et al.*, 2006).

global solution is ensured by an iterative procedure (Vidar, 2011), if global and fast local convergence of this procedure is proven. We consider this procedure as an incremental one because the initial simplified problem P_0 is complicating at each iteration, step-by-step taking into account additional global concerns. In general, the difference between incremental and iterative optimization procedures can be explained in the following way: an incremental procedure starts from an initial simplified problem instance P_0 of the original APSP, constructs a series of its more and more complicated instances $P_0, P_1, \dots, P_N$, finds a feasible solution to each instance P_i, and, starting from a prototype solution x_0 to the problem P and keeping track of current optimum, generates a sequence feasible solutions $x_0, x_1, \dots, x_N$ until the optimal solution to the problem P has been obtained, or, in other words, until P_N becomes equivalent to P (the equivalence between P and P_N optimizers must be shown (Alexandrov and Lewis, 1999)); an iterative procedure starts from an approximate trial solution x_0 to the APSP and generates a sequence gradually refined solutions $x_0, x_1, \dots, x_N$ until a predetermined level of precision has been reached. Thus, an incremental optimization procedure is one, in which "*some of the constraints relax over time. The goal is to find a sequence of feasible solutions, one per time step, such that later solutions build on earlier solutions incrementally*" (Sharp, 2007).

As mentioned above, bi-level decomposition decomposes APSP into a master (or root) problem and relaxed problem which is decomposed into one or more sub-problems. A relaxed problem is one for which: (a) any solution to the APSP corresponds to a feasible solution of relaxed problem; (b) solutions to the relaxed problem find the corresponding solutions to the APSP. It is easier to solve than the original problem because it is smaller and usually has special properties such as convexity, sparsity, or network structure, which can be algorithmically exploited (de Miguel, 2011). Examples are APSP with the constraint matrix of bordered block-diagonal or staircase form. The recent is common in multi-period and in multi-stage planning, including the stochastic planning approaches (Lübecke, 2010). The APSP is relaxed by deleting global concerns. Deleted concerns are left in the master problem, but are defined implicitly, for example, by the set of basic feasible solutions and unbounded directions of the sub-problems (Tebboth, 2001). Local variables and constraints are kept within corresponding sub-problems. According to Gunnerud (2011), "*[t]here are two aspects to consider when choosing which constraints to place in the sub-problem. It should be easy to solve since it is re-optimized several times, and it should provide good quality bounds on the intermediate solutions*". In this way the decomposition separates global and local concerns. For example, enterprise-wide constraints can be separated from shop floor level constraints or coordination between machines constraints can be separated from sequencing constraints that restrict the construction of a schedule for each machine (Gelinas and Soumis, 2006). The master problem coordinates the sub-problems and ensures that the global constraints are satisfied (Tebboth, 2001). In some respects, it is equivalent to the original APSP, but it gives better bounds when its relaxation is solved than when the relaxation of the original APSP is solved (Vanderbeck and Wolsey, 2010). Its "*objective and/or constraint functions are obtained using information gathered at subproblems solutions. At each iteration of the optimization algorithm solving the master problem, all of the N subproblems are solved and information is exchanged* *between the master problem and sub-problems*" (de Miguel, 2011). Usually by solving the master problem we obtain a solution to the original problem (Tebboth, 2001). However, the existence of globally and fast locally convergent optimization algorithms should be proven for the proposed master problem as well as for its sub-problems (de Miguel, 2011). It is possible only in cases when APSP has some properties (e.g. non-degeneracy, smoothness) and when these properties are also preserved in the proposals master problem (Murray and Prieto, 1995). There are a number of propositions how to construct such master problems. If the sub-problems are solved using heuristics, the solution to the APSP may be not necessary optimal. Constraint programming, genetic algorithms and other heuristic approaches can be used to solve sub-problems. In some cases, sub-problems can be decomposed further applying any kind of the above described decomposition approaches.

The bi-level decomposition allows us to solve some kinds of APSP in a decentralized or distributed fashion what leads to a significant simplification of the solution procedure.

The general bi-level decomposition procedure for the APSP optimization can be described in a following way:

1. to relax the APSP, i.e. to drop global issues;
2. to decompose the relaxed problem into sub-problems;
3. to abstract from APSP an easy solvable master problem (initial approximation of APSP), to prove its equivalency to the original APSP including a preservation of properties that ensure global and fast local convergence, to chose a coordination method;
4. to restrict the initial master problem (it is necessary because the initial master problem includes all global concerns that are defined implicitly and for this reason it is far more large than the original problem);
5. to solve the restricted master problem and consider the obtained result as a first "prototype" solution x_0 to the APSP;
6. using the "prototype" solution, to solve all sub-problems;
7. using the solutions to sub-problems as the feedback, to complicate the current master problem (i.e. to include in the master problem some additional global concerns or, in other words, to improve the approximation of the APSP);
8. to solve a new master problem and check a new "prototype" solution on the optimality;
9. if the predefined approximation ratio is achieved, the "prototype" solution is to be considered as the solution to the whole APSP, else go to the step 6.

In an ideal case, the process converges in the sense that the final solution is best one for the master problem as well as for any sub-problem, however, not necessary the "best" always means the "optimal". Even worse, some of bi-level decomposition algorithms may fail to converge even in cases when the starting point is very close to the optimizer (Demiguel and Nogales, 2005).

Due to repetition, bi-level decompositions are less efficient than direct methods. Instead of solving the original APSP with global concerns, there are solved repetitively two problems: a master problem (a simple one) and a relaxed problem (all sub-problems). For LP problems, except some special cases, this approach usually is inefficient but often it is worth to be used to many other convex optimization problems including MIP problems.

There are two main kinds of bi-level decomposition: primal and dual. Primal decomposition decomposes original (primal) problem; dual decomposition decomposes its dual – mostly, Lagrangian dual – problem. If global concern is interpreted as a re-source, then in the primal decomposition the master problem allocates to each sub-problem an amount of resource and in the dual decomposition sets a price for the re-sources. In the recent case, the sub-problem depending on the price decides itself what amount of resource it requires (Chiang *et al.*, 2007). Yet, in the primal decomposition, the master problem is responsible for the proper allocation of the global resources and in the dual de-composition – for producing the best pricing strategy. Primal decomposition is preferred when global concerns are stated as global variables; a dual decomposition – when they are stated as global constraints. However, direct primal and dual decompositions are not always possible. So, sometimes more sophisticated decomposition techniques should be used, for example, the number of dual variables firstly may be reduced by reparameterization and then the dual bi-level decomposition can be done.

Classical examples of bi-level decomposition are the Dantzig–Wolfe decomposition (DWD) (Gunnerud *et al.*, 2010; Pittman *et al.*, 2007; Shapiro, 1993; Dantzig and Wolfe, 1960), and the Benders decomposition (Benders, 1962; Floudas, 1995; Balas, 1983; Taskin, 2010; Olsen, 2012; Aardal and Larsson, 1990). They both have a number of modifications including the reduced variant of DWD (RDWD) (Stadtler *et al.*, 2014), the DWD versions for a general distributed computing environment (Lyu *et al.*, 2004; Rios and Ross, 2010), the generalized Benders decomposition (Geoffrion, 2006), and the Logic-Based Benders Decomposition (LBBD) (Hooker and Ottosson, 2003; Hooker, 2007). In the context of production planning and scheduling, the Dantzig–Wolfe decomposition is also known as a price-directed decomposition and the Benders decomposition – as a resource-directed decomposition (Shapiro, 1993). The first is a special case of the column generation approach and the second is a special case of the row generation approach. In mathematical programming theory, both approaches are already well-investigated and understood. For solving linear problems, the classical DWD "*did not perform better than the Simplex method*" and, consequently, is not competitive, but it "*is a real winner in the context of integer programming*" (Lübecke, 2010).

Let N_t be a set of sub-problems of an initial set N of all sub-problems. In case when, applying the classical DWD, at the iteration t only sub-problems from N_t satisfy the optimality condition, the step 7 of the above described algorithm includes additional global variables into the basis of reduced master problem by including columns for every from N_t sub-problems. The RDWD reduces the number of sub-problems by updating the coefficients of the reduced master problem. As a result, the number of iterations decreases, because the RDWD, differently from classical one, does not compute the optimality condition for every sub-problem at the same time, but stops computing the optimal solution for a sub-problem when this satisfies the optimality condition, even if the other sub-problems do not provide an optimal solution (Stadtler *et al.*, 2014). Consequently, the solution is suboptimal. On the other hand, this approach for LP problems does not affect the convergence. So, computing the suboptimal solution guarantees the feasibility and stability (Stadtler *et al.*, 2014).

LBBD generalize the classical notion of duality and defines an inference dual problem for any optimization problem (Hooker and Ottosson, 2003). An inference dual problem is solved by proving the optimality using an appropriate logical formalism. Solutions to sub-problems are obtained by determining the conditions under which the proof remains valid. LBBD is widely used to solve a variety of planning and scheduling problems. For a more on these applications the reader is referred to Hooker (2007).

In summary, a design of an ensemble of collaborating algorithms which inter-act according to the scheme provided by an incremental collaboration pattern is a very complex task. Usually there are several different ways to decompose a given APSP. Different bi-level decompositions vary in efficiency, robustness, and other characteristics (see Table 1). Even for a given decomposition, a master problem can be constructed differently. In addition, different sub-problems solution methods, including constraint-based approaches, simulation and other heuristics or even meta-heuristics, can be chosen. Finally, a feasibility study should be done and implementability of the ensemble should be investigated in detail.

4.4. *Iterative Collaboration Pattern*

An iterative collaboration pattern (Dauzere-Peres and Lasserre, 2003; Helber and Sahling, 2010; Almeder, 2010) decomposes the APSP into two sub-problems, mostly into planning and scheduling sub-problems. However, the philosophy behind the iterative collaboration pattern essentially differs from the philosophy behind the hierarchical or behind the incremental collaboration patterns. The sub-problems interaction scheme provided in this pattern is similar to the activities interaction scheme defined by an iterative (evolutionary) systems life cycle model (May and Zimmer, 1996; Patton and Jayaswal, 2006). However, typically the pattern defines only two-stage iterative process. It is assumed that a complex interplay there is between both sub-problems. Differently than in the hierarchical pattern, these sub-problems are not considered as higher (master) level and lower (slave) level sub-problems but as two peer partners both equally contributing to the overall APSP solution process. An iterative process starts with solving the first sub-problem. This sub-problem is solved under assumption that the other sub-problem is already solved and its solution is already known. An optimal solution of the first sub-problem is produced under this assumption. The obtained solution is an input of the second sub-problem. If when solving this sub-problem the assumption done in the first sub-problem is accepted, the solution process is finished. Otherwise a new iteration of the solution process starts. The solution of the second sub-problem becomes an input to the first sub-problem and its solution again becomes an input to the second sub-problem. Thus, the pattern generates a sequence of improving approximate solutions of the overall APSP until some convergence criterion is satisfied and an optimal or near-optimal solution is produced. Of course, the APSP should be decomposed into sub-problems in such a way that a sequence of the approximate solutions would be converging to a stable solution. For example, in Dauzere-Peres and Lasserre (2003) the planning sub-problem is modeled using continuous variables, whereas the scheduling sub-problem is modeled using discrete variables. The planning

Table 1
Most important bi-level decomposition approaches.

	Applicability	Efficiency	Convergence
Benders decomposition (Benders, 1962)	To problems with global variables whose objective and constraint functions are linear in the local variables. Is widely used in the APS systems. Requires "*that the sub-problem be a continuous linear or nonlinear programming problem*" (Hooker, 2007).	Inefficient for LP problems, worth to be used for MIP problems Under typical conditions, is globally and fast locally convergent.	
Generalized Benders decomposition (Geoffrion, 2006)	To problems whose objective and constraint functions are convex in the local variables. Is widely used in the APS systems.	Efficient only when the objective and constraint functions are separable with respect to global and local variables.	Is globally and fast locally convergent to approximate solution.
Logic-Based Benders decomposition (Hooker and Ottosson, 2003)	To any class of optimization problems, but a proof scheme and sub-problems solution method must be devised for each class. Enables generic solvers to be used as sub-problem solvers. Is widely used in the APS systems. Provides a framework for combining MILP and constraint programming (CP) techniques. However, "*provides no standard scheme for generating Benders cuts*" (Hooker, 2007).	It is possible to use special structure of the sub-problems, what is impossible in the traditional and generalized Benders decomposition approaches. Due to this possibility, a solution of sub-problems can be significantly more efficient. However, techniques to solve master problem and sub-problems are specific for each class of optimization problems.	Convergence rate depends on a problem structure (Hooker, 2005) and may be unpredictable. Integrated MILP and CP framework obtains probably optimal solutions. The algorithm "*can be terminated at almost any point with a feasible solution and a lower bound on the optimal value*" (Hooker, 2007).
Dantzig–Wolfe decomposition (Dantzig and Wolfe, 1960)	To LP, IP and MILP problems with global constraints. There are several methods to generalize DWD to MILP. For LP problems, except solving in parallel environment, is not competitive with the standard Simplex method, because the DWD suffers from the tailing of effect (slow convergence at the end of the process). DWD can be combined with heuristics, which can "be used to construct or improve primal and dual solutions as often as it seems useful" (Lübecke, 2010).	Sub-problems are mutually independent LP problems, which can be solved in parallel. The parallelism is coarse grained and ideally suited to networks of serial computers. Considering that the work to solve the sub-problems is not dominated by the work to solve the master problem, this parallelism is likely to be efficient (Tebboth, 2001). The algorithm can be speed up solving sub-problems by heuristics.	Is globally and locally convergent. For convex problems, if there exists an initial restricted master problem with a feasible LP relaxation. Due to the accessibility to a dual bound of master problem, it is possible to terminate algorithm, when desired approximation ratio is reached (*early termination*) (Lübecke, 2010).
Tammer's decomposition (Tammer, 1987)	To non-degenerate[a] non-convex problems with global constraints. The usage in the APS systems is unknown.	Algorithm fails when it comes to a value of global variables for which one of the sub-problems is infeasible.	Global convergence is not proven. Fast local convergence under restrictive non-degeneracy assumption.

(*continued in next page*)

Table 1
(*Continued*)

	Applicability	Efficiency	Convergence
Collaborative optimization (Braun, 1996)	To non-convex problems with global variables. Objective functions in sub-problems should be quadratic penalty functions.	It is required to solve master problem for a sequence penalty parameters which tend to infinity. For large values of this parameter, inefficiency may be expected (Murray, 1971).	Global convergence cannot be proven. Fast local convergence if objective functions in all sub-problems are reformulated and master problem becomes smooth in the neighborhood of the minimizer (de Miguel, 2011).

[a] An n-dimensional problem is non-degenerate if the number of active (linearly independent) constraints is not greater than n (Bertsimas and Tsitsiklis, 1997).

algorithm directly considers the sequencing decisions while computing the lot sizes. It solves the planning sub-problem for a fixed sequence of operations on the machines. This sequence is generated by an internal procedure of the planning algorithm. This procedure may implement any scheduling algorithm but this algorithm should be consistent with the main scheduling algorithm used to solve the scheduling sub-problem. The scheduling algorithm solves the scheduling sub-problem for the fixed sizes of lots. To improve the production plan, the algorithm aims to find a sequence of operations on machines which is better than what that was assumed by planning algorithm. The convergence criterion is "*that all jobs end on time and that operations end strictly before the end*" (Dauzere-Peres and Lasserre, 2003) of the period of time associated with these jobs. The planning sub-problem is modeled as a LP problem. The scheduling sub-problem is considered as combinatorial optimization problem. The iterative process searches not for an optimal plan and schedule, but for a globally optimal and feasible production plan.

There is proposed a great number of different variants of iterative pattern. Often it is used to combine heuristic optimization method with mathematical programming procedure which iteratively enhances an initial feasible solution find by this method. For example, in Almeder (2010) the pattern combines an ant-based algorithm with an algorithm for exact solution of mixed-integer linear programs. The hybrid algorithm is designed to solve small and medium-sized problems to determine production schedules as part of the material requirements planning (MRP) process. In Toledo *et al.* (2011) the pattern combines a multi-population genetic algorithm and fix-and-optimize heuristic procedure. The hybrid algorithm is designed to solve the Multi-Level Capacitated Lot Sizing Problem (MLCLSP). The GA defines binary variables for MLCLSP formulation and the best solution found (individual) is also improved by a fix-and-optimize heuristic.

5. Conclusions

The recessions of recent decades, rapidly evolving and changing markets, permanently growing competitive pressures, increasing competition for resources, limited capital budget, increasingly complex regulatory environments, and dramatic increase in capital expenditures and operating costs force the industry to continuingly reduce production timelines and costs without loss in quality, and to improve plant flexibility. The traditional ERP-based Enterprise Information Systems (EIS) with their focus on transaction processing related business aspects cannot cope with these problems. They paid limited attention to the integration of planning and scheduling activities, cannot simultaneously take into account the constraints at both enterprise and plant levels, consider materials and capacity issues together, integrate production, distribution and logistics management issues, and even more-to prepare optimal or near optimal plans and schedules. The concept of EIS should be rethought. Namely, the optimization of plans and schedules for the variety of enterprise's business and manufacturing activities, and the maintenance of their optimality taking into account the permanently changing external and internal state of affairs should be considered as the main goals of EIS. All the more reason for this is that advances in computing technology and in data processing methods and optimization techniques make the optimization of planning and scheduling problems feasible. It means that advanced planning and scheduling (APS) system, supported by one or more ERP systems acting as sub-ordinate transaction-processing systems, should become the dominant component of EIS. The shift from ERP-dominant EISs to APS-dominant EIS is already underway more than ten years.

Despite the relatively long history APS systems, the theory of these systems is still rather in rudimentary phase. Although there is a rich body of literature on APS-oriented optimization models as well as, methods and procedures exists, a theoretical framework allowing the design of an ensemble of collaborating algorithms for a particular APS system is still lacking. The specific APS systems modeling and implementation problems of also are only poorly examined. For example, a significant limitation of current APS systems is to cope with uncertainties. Another problem-their ability to optimize plans across the trade partners in the supply chain-still remains a great challenge. All such problems should be investigated in more detail, theoretically sound solutions of these problems should be designed and incorporated into the main theory of information systems engineering. In our opinion, it is possible to develop the general theory of APS systems because they form a family of congenerous systems even if they are industry-specific and differ in many aspects including planning concepts, tasks, methods and optimization procedures. This family is characterized by shared commonalities, namely, business objectives, problem domains, typical problems, and typical features.

The present paper briefly discusses and assesses the current state of the field and sketches the vision of general APS systems theory. It formulates the research questions that should be answered by this theory and investigates how to design the possibly best ensemble of collaborating algorithms required for a particular APS system. Special attention is paid to the separation concerns problem and to the decomposition of integrated planning and scheduling optimization problem into a complex of sub-problems of manageable sizes. The paper examines the hierarchical, incremental, and iterative collaboration patterns from the design of an ensemble of collaborating algorithms point of view. The main idea suggested is that APS systems should be designed by combining the collaboration patterns and implemented by an ensemble of collaborating algorithms. The quality of the ensemble should be assessed by applying efficiency, finiteness, correctness, robustness, and implementability criteria.

Acknowledgments. The research has been supported by the EUROSTARS Project E!6232-PEN "Production effectiveness navigator". The authors cordially thank project partners Ersan Gürdoğn (HISBIM, Turkey), Piotr Podgórni and Tomasz Woch (IT Vision, Poland), dr. Sándor Vincze (Next System Ltd., Hungary) for detailed and fruitful discussions on the project challenges. Special thanks to Taşkın Kizil (HISBIM, Turkey), who initiated the project and formulated its goals.

References

Aardal, K., Larsson, T. (1990). A Benders decomposition based heuristic for the hierarchical production planning problems. *European Journal of Operational Research* 45(1), 4–14.

Adda, M., Mcheick, H., Mili, H. (2010). Formal model and DSL for separation of concerns based on views. *Journal of Object Technology*, 9(6), 25–50.

Aghezzaf, E.H. (2010). Models for robust tactical planning in multi-stage production systems with uncertain demands. *Computers and Operations Research*, 37(5), 880–889.

Aghezzaf, H., Sitompul, C., van den Broecke, F. (2011). A robust hierarchical production planning for a capacitated two-stage production system. *Computers & Industrial Engineering*, 60(2), 361–372.

Aldrich, J. (2000). Challenge problems for separation of concerns (online publication). In: *Presented at the Workshop on Advanced Separation of Concerns of the 2000 ACM SIGPLAN Conference on Object-Oriented Programming Systems, Languages, and Applications (OOPSLA 2000)*, Mineapolis, USA, October 15–19, 2000. University of Washington. Accesible at http://www.cs.cmu.edu/ aldrich/papers/challenge.pdf.

Al-E-Hashem, S.M.J., Malekly, H., Aryanezhad, M.B. (2011). A multi-objective robust optimization model for multi-product multi-site aggregate production planning in a supply chain under uncertainty. *International Journal of Production Economics*, 134(1), 28–42.

Alexandrov, N., Lewis, R.M. (1999). Comparative properties of collaborative optimization and other approaches to MDO. In: *Proceedings of the 1st ASMO UK/ISSMO Conference on Engineering Design Optimization*, July 8–9, 1999. MCP Press, pp. 1–8.

Almeder, Ch. (2010). A hybrid optimization approach for multi-level capacitated lot-sizing problems. *European Journal of Operational Research*, 200(2), 599–606.

Alvarez, E. (2007). Multi-plant production scheduling in SMEs. *Robotics and Computer-Integrated Manufacturing* 23(6), 608–613.

Ansari, F., Fathi, M., Seidenberg, U. (2013). Combining synoptic and incremental approaches for improving problem-solving in maintenance planning, monitoring and controlling. In: *9th Interdisciplinary Workshop on Intangibles, Intellectual Capital and Extra-Financial Information*, Copenhagen, Denmark, September 26–27, 2013. Workshop papers (CD-ROM). EIASM, Brussels. http://www.wiwi.uni-siegen.de/wiwi/prod/downloads/ansari_fathi_seidelberg.pdf.

Axsater, S., Jonsson, H. (1984). Aggregation and disaggregation in hierarchical production planning. *European Journal of Operational Research* 17(1), 338–350.

Aziz, K. (2002). *Oracle Advanced Planning and Scheduling: Implementation and User's Guide*, Vols. 1 & 2. Release 11i, Part No. B10144-01. Oracle Corporation.

Balas, A. (1983). Benders's method revisited. *Journal of Computational and Applied Mathematics*, (9), 3–12.

Bassil, Y. (2012). A simulation model for the waterfall software development life cycle. *International Journal of Engineering & Technology (IJET)*. 2(5), 1–7.

Baumann, M., Dimitrov, T. (2008). Hierarchical models in advanced planning and scheduling. *Automatisierungstechnik*, 56(2), 90–97.

Behdani, B., Lukszo, Z., Adhitya, A., Srinivasan, R. (2010). Agent-based modeling to support operations management in a multi-plant enterprise. *International Journal of Innovative Computing, Information and Control* 6(7), 2873–2884.

Benders, J.F. (1962). Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematic*, (4), 238–252.

Bertsimas, D., Brown, D.B., Caramanis, C. (2011). Theory and applications of robust optimization. *SIAM Review* 53(3), 464–501.

Bertsimas, D., Tsitsiklis, J.N. (1997). *Introduction to Linear Optimization*. Athena Scientific, Belmont, MA, USA.

Bitran, G.R., Hax, A.C. (1981). Disaggregation and resource allocation using convex knapsack problems with bounded variables. *Management Science*, 27(4), 431–441.

Bitran, G.R., Tirupati, D. (1989). *Hierarchical production planning*. Working paper #3017-89-MS. Massachusetts Institute of Technology/University of Texas, Cambridge/Austin, USA.

Blackstone Jr., J.H. (Ed.) (2010). *APICS Dictionary*, 13th edn. APICS The Association for Operations Management, Chicago, IL, USA.

Borrett, J., Tsang, E. (2009). Adaptive constraint satisfaction: the quickest first principle. In: Mumford, Ch.L., Jain, L.C. (Eds.), *Computational Intelligence, Intelligent Systems Reference Library, 1*. Springer, Berlin/Heidelberg, pp. 203–230.

Braybrooke, D., Lindblom, C. (1963). *A Strategy of Decision: Policy Evaluation as a Social Process*. Free Press, New York.

Braun, R.D. (1996). *Collaborative optimization: an architecture for large-scale distributed design*. PhD thesis. Stanford University, Stanford, CA, USA.

Bubenik, P. (2011). Advanced planning system in small buisiness. *Applied Computer Science*, 7(2), 21–26.

Cabantous, L., Gond, J.-P. (2011). Rational decision making as performative praxis: explaining rationality's éternel retour. *Organization Science*, 22(3), 573–586.

Caplinskas, A., Dzemyda, G., Kiss, F., Lupeikiene, A. (2012). Processing of undesirable business events in advanced production planning systems. *Informatica* 23(4), 563–579.

Castillo, E., Minguez, R., Castillo, C. (2008). Sensitivity analysis in optimization and reliability problems. *Reliability Engineering and System Safety*, 93(12), 1788–1800.

Chang, T.C., Wysk, R.A. (1985). *An Introduction to Automated Process Planning Systems*. Prentice-Hall, Inc., New York.

Chen, P., Ji, P. (2007). A mixed integer programming model for advanced planning and scheduling (APS). *European Journal of Operational Research* 181, 515–522.

Chen, Sh.-Ch., Kao, Y.-Ch, Chen, H.-H. (2012). Advanced planning and scheduling (APS) and enterprise planning system (ERP) planning reference models for Personal Computer assembly industry: a process perspective. *African Journal of Business Management*, 6(14), 4784–4794.

Chiang, M., Low, S.H., Calderbank, A.R., Doyle, J.C. (2007). Layering as optimization decomposition: a mathematical theory of network Architectures. *Proceedings of the IEEE*, 95(1), 255–312.

Conejo, A.J., Castillo, E., M?nguez, R., Garc?a-Bertrand, R. (2006). *Decomposition Techniques in Mathematical Programming: Engineering and Science Applications*. Springer, Berlin/Heidelberg.

Consortium PSLX (2005). *Advanced Planning and Scheduling (APS) Conceptual Definition and Implementation*. White paper, 2005. Accessed: January 2014, available at: http://www.pslx.org/en/doc/2005/WP-01EN-ALL.pdf.

Cote, G., Laughton, M. (1984). Large-scale mixed integer programming: benders-type heuristics. *European Journal of Operational Research*, 16(3), 327–333.

Dantzig, G.B., Wolfe, P. (1960). Decomposition principle for linear programms. *Operation Research*, 8(1), 101–111.

Dauzere-Peres, S., Lasserre, J.B. (2003). On the importance of sequencing decisions in production planning and scheduling. *Transactions in Operational Research*, 9(6), 779–793.

de Carvalho, M.F.G., Haddad, R.B.B. (2012). Production scheduling on practical problems. In: Righi, R. (Ed.), *Production Scheduling*, InTech, Rijeka, Croatia, pp. 157–182.

de Miguel, A.-V. (2011). *Two decomposition algorithms for nonconvex optimization problems with global variables*. PhD thesis. Stanford University, Stanford, CA, USA.

de Santa-Eulalia, L.A., D'Amours, S., Frayret, J.-M., Menegusso, C.C., Azevedo, R.C. (2011). Advanced supply chain planning systems (APS) today and tomorrow. In: Onkal, D. (Ed.), *Supply Chain Management Pathways for Research and Practice*, InTech, Rijeka, Croatia, pp. 171–200.

Demiguel, A.-V., Nogales, F.J. (2005). On the relationship between bilevel decomposition algorithms and direct interior-point methods. In: *Statistics and Econometrics Working Papers* (presented at the eight SIAM Conference on Optimization, Stockholm, Sweden, May 2005). Online publication: http://www.optimization-online.org/DB_FILE/2004/04/865.pdf. Universidad Carlos III.

Dietterich, T. (2000). Ensemble methods in machine learning. In: Kittler, J., Roli, F. (Eds.), *Multiple Classifier Systems: Proceedings of the First International Workshop (MCS 2000)*, Cagliari, Italy, June 21–23, 2000, LNCS, Vol. 1857. Springer, Berlin/Heidelberg, 1–15.

Dijkstra, E.W. (1982). On the role of scientific thought. In: *Writings on Computing: A Personal Perspective*. Springer, New York, pp. 60–66.

EyeOn (2014). *Advanced planning and scheduling in the heigh tech industry*. White paper. Accessed: January 2014, available at: http://www.eyeon.nl/documenten/whitepapers/eyeon_wp_advanced_planning_and_scheduling_ht.pdf.

Epstein, L., van Stee, R. (2008). Approximation schemes for packing splittable items with cardinality. In: Kaklamanis, Ch., Skutella, M. (Eds.), *Approximation and Online Algorithms: 5th International Workshop (WAOA 2007), Eilat, Israel, October 2007*, Revised papers, LNCS, vol. 4927. Springer, Berlin/Heidelberg pp. 232–245.

Esmaeili, A., Jafarnejad, A., Jola, F. (2013). Multi-stage stochastic programming models in production planning. *International Journal of Basic and Applyed Science*, 2(2), 195–200.

Fleischmann, B., Meyr, H., Wagner, M. (2005). Advanced planning. In: Stadtler, H., Kliger, Ch. (Eds.), *Supply Chain Management and Advanced Planning: Concepts, Models, Software and Case Studies*, 3rd edn. Springer, Berlin/Heidelberg, pp. 81–108.

Floudas, Ch.A. (1995). Chapter 6: Mixed-integer nonlinear optimization. In: *Nonlinear and Mixed-Integer Optimization: Fundamentals and Applications*. Oxford University Press, New York/Oxford, pp. 109–138.

Garrido, A., Barber, F. (2011). Integrating planning and scheduling. *Applied Artificial Intelligence: An International Journal*, 15(5), 471–491.

Gelinas, S., Soumis, F. (2006). Dantzig–Wolfe decomposition for job shop scheduling. In: Desaulniers, G., Desrosiers, J., Solomon, M.M. (Eds.), *Column Generation*. Springer, Berlin, pp. 271–302.

Geoffrion, J.M. (1972). Generalized benders decomposition. *Journal of Optimization Theory and Applications*, 10(4), 237–260.

Geoffrion, A.M., Marsten, R.E. (1972). Integer programming algorithms a framework and state-of-the-art survey. *Management Science*, 18(9), 465–491.

Glinz, M. (2007). On non-functional requirements. In: *Proceedings of the 15th IEEE International Requirements Engineering Conference*, Delhi, India, 15–19 October 2007. IEEE Press, New York, pp. 21–226.

Gnonia, M.G., Iavagnilioa, R., Mossaa, G., Mummoloa, G., Di Leva, A. (2003). Production planning of a multi-site manufacturing system by hybrid modeling: a case study from the automotive industry. *International Journal of Production Economics*, 85(2), 251–262.

Graves, S.C. (2011). Uncertainty and production planning. In: Kempf, K., Keskinocak, P., Uzsoy, R. (Eds.), *Planning Production and Inventories in the Extended Enterprise: A State of the Art Handbook, Vol. 1: International Series in Operations Research & Management Science, 151*. Springer, New York, pp. 83–101.

Grossmann, I.E. (2012). Advances in mathematical programming models for enterprise-wide optimization. *Computers and Chemical Engineering*, 47, 2–18.

Gunnerud, V. (2011). *On decomposition and piecewise linearization in petroleum production optimization*. PhD thesis. Norwegian University of Science and Technology, Trondheim, Norway.

Gunnerud, V., Foss, B, Torgnes, E. (2010). Parallel Dantzig–Wolfe decomposition for real-time optimization-applied to a complex oil field. *Journal of Process Control*, 20(9), 1019–1026.

Gupta, V., Langbort, C., Murray, R.M. (2006). On the robustness of distributed algorithms. In: *Proceedings of the 45th IEEE Conference onthe Decision and Control*, 13–15 December 2006, San Diego, CA, USA. IEEE Press, pp. 3473–3478.

Halton, E. (2004). Pragmatism. In: Ritzer, G.(Ed.), *Encyclopedia of Social Theory*. Sage Publications, Thousand Oaks, pp. 596–600.

Hamilton, S. (2014). How do APS and ERP fit together? In: *SearchManufacturing ERP*. Accessed: January 20, 2014, available at: http://searchmanufacturingerp.techtarget.com/answer/How-do-APS-and-ERP-fit-together.

Hax, A.C., Candea, D. (1984). Chapier 6: Hierarchical production planning systems. In: *Production and Inventory Management*. Prentice-Hall, Englewood Cliffs, NJ, 393–407.

Hax, A.C., Meal, H.C. (1975). Hierarchical integration of production planning and scheduling. In: Geisler, M.A. (Ed.), *Studies in Management Scinces. Vol. 1. Logistics*. North Holland/American Elsevier, New York, pp. 53–69.

Hebrard, E. (2007). *Robust solutions for constraint satisfaction and optimization under uncertainty*. PhD thesis. University of New South Wales, New South Wales, Australia.

Helber, S., Sahling, F. (2010). A fix-and-optimize approach for the multi-level capacitated lot sizing problem. *International Journal of Production Economics*, 123(2), 247–256.

Hennet, J.-C. (1999). From the aggregate plan to lot-sizing in multi-level production planning. In: Brandimarte, P., Villa, A. (Eds.), *Modeling Manufacturing Systems: From Aggregate Planning to Real-Time Control*. Springer, Berlin/Heidelberg, New York, pp. 5–23.

Herrmann, J.W. (2006). A history of production scheduling. In: Herrmann, J.W. (Ed.), *Handbook of Production Scheduling. International Series in Operations Research & Management Science, vol. 89*. Springer, New York, pp. 1–22.

Homem de Mello, L.C., Sanderson, A. (1990). AND/OR graph representation of assembly plans. *IEEE Journal of Robotics and Automation*, 6(2), 188–199.

Hooker, J. (2005). Convex programming methods for global optimization. In: Jermann, Ch., Neumaier, A., Sam, D. (Eds.), *Proceedings of the Second International Conference on Global Optimization and Constraint Satisfaction (COCOS'03)*. LNCS, vol. 3478. Springer, Berlin/Heidelberger, pp. 46–60.

Hooker, J. (2007). Planning and scheduling by logic-based benders decomposition. *Operations Research*, 55(3), 588–602.

Hooker, J., Ottosson, G. (2003). Logic-based benders decomposition. *Mathematical Programming*, 96(1), 33–60.

Hopp, M. (2007). *Sales and Inventory Planning with SAP APO*. Galileo Press, Bonn/Boston.

Huberman, B.A., Lukose, R.M., Hogg, T. (1993). An economics approach to hard computational problems. *Science*, 275, 51–54.

Hvolby, H.-H., Steger-Jensen, K. (2010). Technical and industrial issues of advanced planning and scheduling (APS) systems. *Computers in Industry*, 61(9), 845–851.

InQu Informatics (2014). Manufacturing Execution System – InQu.MES. InQu Informatics GmbH, Dresden.

Kallrath, J., Maind, T.I. (2006). *Real Optimizaton with SAP APO*. Springer, Berlinn/Heidelberg.

Kanyalkar, P., Kadil, G.K. (2005). An integrated aggregate and detailed planning in a multi-site production environment using linear programming. *International Journal of Production Research*, 43(20), 4431–4454.

Karimian, P., Herrmann, J.W. (2009). Separating design optimization problems into decision-based design processes. *Journal of Mechanical Design* (online journal), 131(1), 011007.

Kennerley, M., Neely, A. (2001). Enterprise resource planning: analysing the impact. *Integrated Manufacturing System*, 103–113.

Kilger, Ch. (2005). The definition of a supply Chain Project. In: Stadtler, H., Kliger, Ch. (Eds.), *Supply Chain Management and Advanced Planning: Concepts, Models, Software and Case Studies*. 3rd edn. Springer, Berlin/Heidelberg, pp. 281–301.

Kilger, Ch., Wetterauer, U. (2005). The selection process. In: Stadtler, H., Kliger, Ch. (Eds.), *Supply Chain Management and Advanced Planning: Concepts, Models, Software and Case Studies*. 3rd edn. Springer, Berlin/Heidelberg, pp. 303–316.

Kim, H., Lu, J.-C., Kvam, P.H., Tsao, Y.C. (2011). Ordering quantity decisions considering uncertainty in supply-chain logistics operations. *International Journal of Production Economics*, 134(1), 16–27.

Kjellsdotter, L.I. (2009). *Advanced planning and scheduling systems in manufacturing planning processes. L2009:036*. Thesis for the degree of licentiate of engineering, Chalmers University of Technology, Göteborg, Sweden.

Kjellsdotter, L.I. (2012). Shop floor characteristics influencing the use of advanced planning and scheduling systems. *Production Planning & Control: The Management of Operations*, 23(6), 452–467.

Kjellsdotter, L.I. (2012). *Use of advanced planning and scheduling (APS) systems to support manufacturing planning and control processes*, PhD thesis. Chalmers University of Technology, Göteborg, Sweden.

Kjellsdotter, L.I., Jonsson, P. (2010). The potential benefits of APS systems in the sales and operations planning process. *Industrial Management & Data Systems*, 110(5), 659–681.

Kottho, L. (2012). Algorithm selection for combinatorial search problems: a survey. *The Computing Research Repository CoRR*, abs/1210.7959, ACM, New York.

Larman, C., Basil, V.R. (2003). Iterative and incremental development: a brief history. *Computer*, 36(6), 47–56.

Lee, Y.H., Jeong, C.S., Moon, C. (2002). Advanced planning and scheduling with outsourcing in manufacturing supply chain. *Computers & Industrial Engineering*, 43(1–2), 351–374.

Lin, C., Hwang, S., Wang, E. (2007). A reappraisal on advanced planning and scheduling systems. *Industrial Management & Data Systems*, 107(8), 1212–1226.

Lindblom, Ch.E. (1959). The science of muddling through. *Public Administration Review*, 19(2), 79–88.

Lyu, J., Luh, H., Lee, M.-Ch. (2004). Solving linear programming problems on the parallel virtual machine environment. *American Journal of Applied Sciences*, 1(2), 90–94.

Lolidis, M. (2008). *KAIZEN – Definitions & principles on suggestion systems*. Art. No. 001/0708.

Lübbecke, M.E. (2010). Column generation. In: *Wiley Encyclopedia of Operations Research and Management Science (EORMS)*. Willey Online Library. Available at: http://onlinelibrary.wiley.com/.

May, E.L., Zimmer, B.A. (1996). The evolutionary development model for software. *Hewlett-Packard Journal* (August), 1–8.

Malindžák, D., Mervart, J., Lenort, R. (2011). The logistic principles for fast flexible strategy design of the company in crisis time. *Managing Global Transitions*, 9(2), 129–149.

Maravelias, Ch.T., Sung, Ch. (2009). Integration of production planning and scheduling: overview, challenges and opportunities. *Computers and Chemical Engineering*, 33, 1919–1930.

Meeden, L,A. (1994). *Towards planning: incremental investigations into adaptive robot control*. PhD thesis. Indiana University, Bloomington, USA.

Meyr, H., Wagner, M., Rohde, J. (2005). Structure of advanced planning systems. In: Stadtler, H., Kliger, Ch. (Eds.), *Supply Chain Management and Advanced Planning: Concepts, Models, Software and Case Studies*. 3rd edn. Springer, Berlin/Heidelberg, pp. 109–115.

Minslry, M. (1988). *The Society of Mind, First Touchstone edition. A Touchstone Book*. Simon & Schuster, New York/London/Ioronto/Sydney/Tokyo/Singapure.

Myers, K.L., Smith, S.F., Hildum, D.W., Jarvis, P.A., de Lacaze, R. (2001). Integrating planning and scheduling through adaptation of resource intensity estimates. In: Cesta, A., Borrajo, D. (Eds.), *Recent Advances in AI Planning: Proceedings of the 6th European Conference on Planning (ECP-01)*, Toledo, Spain, September, 2001, LNCS. Springer, Berlin/Heidelberg, pp. 1–13.

Moon, Ch., Seo, Y. (2005), Evolutionary algorithm for advanced process planning and scheduling in a multi-plant. *Computers & Industrial Engineering*, 48(2), 311–325.

Moon, Ch., Kim, J., Hur, S. (2002). Integrated process planning and scheduling with minimizing total tardiness in multi-plants supply chain. *Computers & Industrial Engineering*, 43(1–2), 331–349.

Moon, C., Kim, J., Gen, M. (2004). Advanced planning and scheduling based on precedence and resource constraints for e-plant chains. *International Journal of Production Research*, 42(15), 2941–2955.

Mulvey, J.M., Vanderbei, R.J., Zenios, S.A. (1995). Robust optimization of large-scale systems. *Operations Research*, 43(2), 264-281.

Murray, W. (1971). Analytical expressions for the eigenvalues and eigenvectors of the Hessian matrices of barrier and penalty functions. *Journal of Optimization Theory and Applications*, 7(3), 189–196.

Murray, W., Prieto, F.J. (1995). A sequential quadratic programming algorithm using an incomplete solution of the subproblem. *SIAM Journal of Optimization* 5(3), 590–640.

Neureuther, B.D. (2004). Aggregate planning in make-to-order environments (Chapter 002-0326). In: Gupta, S., Garg, N. (Eds.), P*roceedings of the Second World Conference on POM and 15th Annual POM Conference*, Cancun, Mexico, April 30–May 3, 2004. CD:ISSN 1548-4882. European Operations Management Association, Production and Operations Management Society, Japanese Society for Production Management, pp. 1–15.

Nicholas, J. (2011). *Lean Production for Competitive Advantage: A Comprehensive Guide to Lean Methodologies and Management Practices*. CRC Press, New York.

Northover, M., Kourie, D.G., Boake, A., Gruner, S., Northover, A. (2008). Towards a philosophy of software development: 40 years after the birth of software engineering. *Journal for General Philosophy of Science*, 39(1), 85–113.

Oboulhas, C., Xiaofei, X., Dechen, Z. (2005). A model for multi-plant production planning coordination. *The International Arab Journal of Information Technology*, 2(3), 177–182.

Olsen, N.V. (2006). *Incremental product development: Four essays on activities, resources, and actors*. PhD dissertation. BI Norwegian School of Management, Oslo.

Ouerfelli, H., Dammak, A., Kallel Chtourou, E. (2012). Benders-based approach for an integrated Lot-Sizing and Scheduling problem. *International Journal of Combinatorial Optimization Problems and Informatics*, 3(3), 59–68.

Öztürk, C., Ornek, A.M. (2014). Operational extended model formulations for advanced planning and scheduling systems. *Applied Mathematical Modelling*, 38(1), 181–195.

Patton, P.C., Jayaswal, B.K. (2006). Chapter 1: software development methodology today. In: *Design for Trustworthy Software: Tools, Techniques, and Methodology of Developing Robust Software*, 1st edn. Prentice-Hall, New York, pp. 3–33.

Peeters, J. (2009). Early MRP systems at royal Philips electronics in the 1960s and 1970s. *IEEE Annals of the History of Computing*, 31(2), 56–69.

Phanden, R.K., Jain, A., Verma, R. (2013). An approach for integration of process planning and scheduling. *International Journal of Computer Integrated Manufacturing*, 26(4), 284–302.

Pimentel, S.G., Brem, L.M. (1994). Robust planning in uncertain environments. In: *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence (UAI1994)*, pp. 463–469.

Pittman, S.D., Bare, B.B., Briggs, D.G. (2007). Hierarchical production planning in forestry using price-directed decomposition. *Canadian Journal of Forest Research*, 37(10), 2010–2021.

Popper, K. (2003). *Das Elend des Historizismus*. 6 Auf. Mohr Siebeck, Tübingen.

Ptak, C., Smith, Ch. (2008). *Beyond MRP. Meeting the current materials synchronizations chalenges*. White paper. Constraints Management Group, Enumclaw.

Quadt, D., Kuhn, H. (2005). Conceptual framework for lot-sizing and scheduling of flexible flow lines. *International Journal of Production Research*, 43(11), 2291–2308.

Quinn, J. (1980). An incremental approach to strategic change. *The McKinsey Quarterly*, 34–52.

Rice, J.R. (1975). *The algorithm selection problem*. Computer Science Technical Reports 75-152, Purdue University.

Rios, J., Ross, K. (2010). Massively parallel Dantzig–Wolfe decomposition applied to traffic flow scheduling. *Journal of Aerospace Computing Information and Communication*, 7(1), 32–45.

Rohde, J. (2005). Coordination and integration. In: Stadtler, H., Kliger, Ch. (Eds.), *Supply Chain Management and Advanced Planning: Concepts, Models, Software and Case Studies*. 3rd edn. Springer, Berlin/Heidelberg, 245–257.

Royce, W. (1970). Managing the development of large software systems. In: *Proceedings of IEEE WESCON 26*. Institute of Electrical and Electronics Engineers, originaly publshed by TRW, pp. 1–9.

Saad, G.H. (1990). Hierarchical production-planning systems: extensions and modifications. *Journal of the Operational Research Society*, 41(7), 609–624.

Safaei, M.S., Hussein, S.M.M., Farahani, Jolai, F., Ghodsypour, S.H. (2010). Integrated multi-site production-distribution planning in supply chain by hybrid modeling. *International Journal of Production Research*, 48(14), 4043–4069.

Shapiro, J. (1933). Mathematical programming models and methods for production planning and scheduling (Chapter 8). In: Graves, S.C., Rinnooy Kan, A.H.G., Zipkin, P.H. (Eds.) (1993). *Handbooks in Operations Research and Management Science. Logistics of Production and Inventory, vol. 4*. Elsevier Science, Amsderdam, pp. 523–566.

Sharp, A.M. (2007). *Incremental algoritms: solving problems in a changing world*, PhD dissertation. Cornell University, Ithaca, NY, USA

Simon, H. (1997). *Administrative Behavior*. 4th edn. Free Press, New York.

Solberg, A., Simmonds, D., Reddy, R., Ghosh, S., France, R. (2005). Using aspect oriented techniques to support separation of concerns in model driven development. In: *Proceedings of the 29th Annual International Conference on Computer Software and Applications (COMPSAC 2005)*, vol. 1, 26–28 July 2005, Los Alamitos, USA. IEEE, Washington, DC, USA, pp. 121–126.

Stadtler, H. (2004). Supply chain management and adviced planning – basics, overview and chalenges. *European Journal of Operation Research* (163), 575–588.

Stadtler, H., Kilger, C. (Eds.) (2005). *Supply Chain Management and Advanced Planning – Concepts, Models Software and Case Studies*, 3rd edn. Springer, Berlin.

Standardi, L., Poulsen, N.K., Jorgensen, J.B. (2014). A reduced Dantzig–Wolfe decomposition for a suboptimal linear MPC. In: *Preprints of the 19th World Congress of The International Federation of Automatic Control*, Cape Town, South Africa, August 24–29, 2014. IFAC, pp. 2207–2212.

Sung, J., Jeong, B. (2014). An adaptive evolutionary algorithm for traveling salesman problem with precedence constraints. *The Sientific World Journal* (online). http://dx.doi.org/10.1155/2014/313767. Article ID 313767.

Tammer, K. (1987). The application of parametric optimization and imbedding for the foundation and realization of a generalized primal decomposition approach. In: Guddat, J., Jongen, H., Kummer, B., Nozicka, F. (Eds.), *Parametric Optimization and Related Topics. Mathematical Research, Vol. 35*. Akademie-Verlag, Berlin, pp. 376–386.

Tan, W., Khoshnevis, B. (2000). Integration of process planning and scheduling – a review. *Journal of Intelligent Manufacturing*, 11(1), 51–63.

Taskin, Z.C. (2010). Benders decomposition. In: Cochran, J. (Ed.), *Encyclopedia of Operations Research and Management Science*. Wiley, New York, pp. 1–15.

Tebboth, J.R. (2001). *A computational study of Dantzig–Wolfe decomposition*. PhD thesis. University of Buckingham, Buckingham, Buckinghamshire, England.

Tempelmeier, H. (2001). Supply chain planning with advanced planning systems. In: *Proceedings of the 3rd Aegean International Conference on Design and Analysis of Manufacturing Systems*, May 19-22, 2001, Tinos Island, Greece. University of the Aegean, pp. 1–10. https://www.icsd.aegean.gr/aic2001/program.htm.

Texeira Jr., R.F., Coube, L.E.C. (2009). Analysis of the main functionalities of an advanced planning and scheduling system (APS) geared towards small companies. In: Hanna, M.D. (Ed.), *Online Proceedings of the 20th Annual Conference of Production and Operations Management Society (POM 2009)*, Orlando, FL, USA, May 1–4, 2009. http://www.pomsmeetings.org/ConfProceedings/011/FullPapers/Fullpaper.htm.

Theeuwen, M. (2007). *Advanced planning and scheduling in the high tech industry*. An EyeOn white paper. EyeOn, NL-5735 ZH, Aarle-Rixtel.

Toledo, C.F.M., de Oliveira, R.R.R., Franca, P.M. (2011). A hybrid heuristic approach to solve the multi-level capacitated lot sizing problem. In: *Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2011)*, New Orleans, LA, USA, 5–8 June, 2011. IEEE, New York, pp. 1194–1201.

Tovey, C. (2002). Tutorial on computational complexity. *Interfaces*, 32(3), 30–61.

van Landeghem, H., Vanmaele, H. (2002). Robuust planning: a new paradigm for demand chain planning. *International Journal of Operations Management*, 20(6), 769–783.

Vanderbeck, F., Wolsey, L.A. (2010). Reformulation and decomposition of integer programs. In: Jünger, M., Liebling, T.M., Naddef, D., Nemhauser, G.L., Pulleyblank, W.R., Reinelt, G., Rinaldi, G., Wolsey, L.A. (Eds.), *50 Years of Integer Programming 1958–2008*. Springer, Berlin/Heidelberg, pp. 431–502.

Vassilevska, V., Williams, R., Woo, S.L.M. (2006). Confronting hardness using a hybrid approach. In: *Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'06)*, January 22–24, Miami, FL, USA. ACM Press, New York, pp. 1–10.

Wang, J. (2004). A fuzzy robust scheduling approach for product development projects. *European Journal of Operational Research*, 152(1), 180–194.

Wegener, I. (2005). *Complexity Theory: Exploring the Limits of Efficient Algorithms*. Springer, Berlin/Heidelberg.

Vidar, G. (2011). *On decomposition and piecewise linearization in petroleum production optimization*. PhD thesis. Norwegian University of Science and Technology, Trondheim, Norway.

Wortmann, J.C. (1988). Evolution of ERP systems. In: Bititci, U.S., Carrie, A.S. (Eds.), *Strategic Management of the Manufacturing Value Chain: Proceedings of the International Conference of the Manufacturing Value-Chain*, August'98, Troon, Scotland, UK, IFIP 2. Kluwer Academic, Boston, pp. 11–23.

Wu, D.D., Olson, D.L., Birge, J.R. (2011). Introduction to special issue on "Enterprise risk management in operations". *International Journal of Production Economics*, 134(1), 1–2.

Zhang, H., Gen, M. (2006). Effective genetic approach for optimizing advanced planning and scheduling in flexible manufacturing system. In: Cattolico, M. (Ed.), *Proceedings of the Genetic and Evolutionary Computing Conference (GECCO'06)*, July 8–12, 2006, Seattle, Washington, USA. ACM, New York, pp. 1841–1848.

Zijm, W.H.M. (2000). Towards intelligent manufacturing planning and control systems. *OR Spektrum*, 22(3), 313–345.

A. Lupeikiene is an associate professor, researcher at the Institute of Informatics and Mathematics of Vilnius University. Her main research interests include information systems, information systems engineering, and decision making.

G. Dzemyda is a member of Lithuanian Academy of Sciences, professor, principal researcher and director of the Institute of Informatics and Mathematics of Vilnius University. His main research interests include optimization and visualization, data mining, and medical informatics.

F. Kiss is an associate professor at the Budapest College of Communication and Business. He also is R& D Director of the Infota Research Institute (Hungary), director of International McLeod Institute of Simulation Sciences Hungarian Satellite Center at Infota, member of the Editorial Advisory Board of the Journal of Modelling in Management. His main research interests include information and knowledge manage-ment methodologies, processes and applications, financial IT systems and decision support.

A. Caplinskas is a professor, chief specialist at the Software Engineering Department of the Institute of Informatics and Mathematics of Vilnius University. His main research interests include enterprise engineering, software engineering, and service engineering.

Moderniosios planavimo ir tvarkaraščių sudarymo sistemos: modeliavimo ir realizavimo problemos

Audronė LUPEIKIENĖ, Gintautas DZEMYDA, Ferenc KISS, Albertas CAPLINSKAS

Straipsnis apibendrina pastarųjų 20 metų tyrimus moderniųjų planavimo ir tvarkaraščių sudarymo (PTS) sistemų modeliavimo ir realizavimo srityje. Jame aptariama šiuolaikinė tokių sistemų koncepcija ir išryškinamos modeliavimo ir realizavimo problemos, kurias tenka spręsti jų kūrėjams. Dalis šių problemų suformuluota analizuojant literatūrinius šaltinius, kitos – apibendrinant autorių patirtį, įgytą kuriant pramoninio pobūdžio sistemą „Gamybos efektyvumo navigatorius“. Darbo mokslinis naujumas yra bendradarbiaujančių algoritmų ansamblio samprata. Tai svarbus indėlis į besiformuojančią PTS sistemų teoriją.