International Journal of Production Research

ISSN: (Print) (Online) Journal homepage: www.tandfonline.com/journals/tprs20

An LSTM network-based genetic algorithm for integrated procurement and scheduling optimisation

Alexander Bubak, Benjamin Rolf, Tobias Reggelin, Sebastian Lang & Heiner Stuckenschmidt

To cite this article: Alexander Bubak, Benjamin Rolf, Tobias Reggelin, Sebastian Lang & Heiner Stuckenschmidt (03 Dec 2024): An LSTM network-based genetic algorithm for integrated procurement and scheduling optimisation, International Journal of Production Research, DOI: 10.1080/00207543.2024.2434948

To link to this article: https://doi.org/10.1080/00207543.2024.2434948

Published online: 03 Dec 2024.

INTERNATIONAL JOURNAL OF PRODUCTION RESEARCH
https://doi.org/10.1080/00207543.2024.2434948

Taylor & Francis
Taylor & Francis Group

An LSTM network-based genetic algorithm for integrated procurement and scheduling optimisation

Alexander Bubak [a], Benjamin Rolf [b], Tobias Reggelin [b], Sebastian Lang [b,c] and Heiner Stuckenschmidt [a]

[a]Fakultät für Wirtschaftsinformatik und Wirtschaftsmathematik, Universität Mannheim, Mannheim, Germany; [b]Institut für Logistik und Matrialflusstechnik, Otto-von-Guericke-Universität Magdeburg, Magdeburg, Germany; [c]Fraunhofer-Institut für Fabrikbetrieb und -automatisierung IFF, Magdeburg, Germany

ABSTRACT

Modern supply chains are characterised by high complexity, requiring effective management through coordinated activities across interrelated functions. This study aims to move from isolated optimisation to integrated decision-making, which offers new potential for efficiency. We investigate an integrated procurement-production problem based on a real case study from a German company specialising in printed circuit board assembly. We propose a novel solution approach that combines a genetic algorithm with a neural network to increase computational efficiency. Our comprehensive evaluation scheme demonstrates the viability of the approach in generating integrated decisions within a limited time frame. Specifically, we quantify the benefits of integrated over separated decision-making at the operational level, extending previous research focussed on the tactical level. The results indicate considerable benefits of integrated decision-making across a wide range of cost factors, although the exact savings depend on specific cost parameters. In addition, we evaluate our model on a rolling horizon planning basis, which is crucial for modelling realistic supply chain behaviour and remains underrepresented in the literature.

ARTICLE HISTORY
Received 15 May 2024
Accepted 19 November 2024

KEYWORDS
Supply chain management; integrated procurement production problem; hybrid flow shop scheduling; genetic algorithm; supervised learning; rolling horizon planning

ABBREVIATIONS
GA: Genetic algorithm; LSTM: Long short-term memory; MILP:Mixed-integer linear program; OAP: Order allocation problem; OR: Operations research; PCB: Printed circuit board; RNN: Recurrent neural network; TS: Tabu search; VNS:Variable neighbourhood search

1. Introduction

In supply chain management, the integration of procurement, production and distribution operations is critical to improving efficiency and market responsiveness. Traditional approaches that manage these functions separately often fail to recognise their interdependencies, leading to inefficiencies such as increased costs and delays due to poor coordination between stages (Darvish and Coelho 2018; Shirvani and Shadrokh 2013). Historically, operations research (OR) in supply chains has focussed on the isolated optimisation of specific operational problems, such as scheduling or vehicle routing. This focus has been driven by the complexity of these problems and the limitations of computational power. However, advances in computing power and the advent of machine learning algorithms are gradually diminishing these limitations. Integrated planning is emerging as a promising approach to exploit new potentials in operational supply chain planning (Darvish and Coelho 2018). These advances allow for more comprehensive models that can consider multiple stages of the supply chain simultaneously, improving overall efficiency and responsiveness. Despite the demonstrated benefits of integration at the tactical planning level, evidence at the more complex operational level remains scarce (Hrabec, Magnus Hvattum, and Hoff 2022; Thevenin, Zufferey, and Glardon 2017). In particular, studies on the integration of procurement and production are still underrepresented in the literature. This gap highlights the need for further research on integrated planning approaches at the operational level, with a particular focus on procurement and production.

This article addresses the integration of procurement and production decisions by considering the case of a medium-sized German company specialising in the assembly of printed circuit boards. The company faces significant challenges in delivering products to customers on time, primarily due to poor scheduling decisions

CONTACT Alexander Bubak alexander.bubak@uni-mannheim.de

and raw material shortages caused by uncertain market environments. These issues underscore the need for a more integrated approach to planning. The main objectives of this study are to develop an efficient solution approach to tackle the complex integrated procurement-production problem and to quantify the benefits of integrated planning compared to separated planning. This research specifically focuses on the procurement and production stages under conditions of stochastic demand, excluding the distribution stage for the present study. The research questions guiding this study are:

(1) How can we integrate operational procurement and production decisions efficiently?
(2) What is the value of integrated planning in procurement and production compared to separated planning?

To investigate the impact of integrated planning, we generate several problem instances with varying sizes, planning approaches and cost scenarios based on the real-world case. This approach allows for a comprehensive empirical evaluation of various factors influencing the performance of both integrated and separated solution approaches. To satisfy the efficiency requirements of operational planning, we present a novel integrated algorithm based on a GA combined with a neural network. The neural network is trained to predict the quality of procurement plans based on production schedules, thereby accelerating the solution approach.

The remainder of this article is structured as follows. Section 2 reviews the relevant literature on integrated supply chain planning. The procurement and production problems considered in this article are formulated in Section 3. In Section 4, we present three separated and two integrated baseline methods as well as our novel solution approach. Empirical findings are presented in Section 5. Section 6 extracts managerial insights from these results. Finally, we conclude by answering the research questions and presenting opportunities for future research in Section 7.

2. Literature review

Today's supply chains are often highly complex, requiring effective management to coordinate activities across multiple interrelated levels and functions. This management has to deal with many factors, such as fluctuating demand and unexpected disruptions. The trend towards globalisation further increases the vulnerability of supply chains (Javorcik 2020). Given these challenges, planning an entire supply chain within a single, monolithic system is impractical (Fleischmann, Meyr, and Wagner 2008). A common approach is to divide the planning process into distinct, manageable modules (Shirvani and Shadrokh 2013). Despite treating these modules as separate entities, the complexity of typical planning problems, including order allocation, scheduling, and vehicle routing, has long been a significant challenge for researchers. Integrated planning is emerging as a promising strategy for improving operational efficiency, offering the potential for greater savings compared to optimising individual functions in isolation (Moons et al. 2017). Therefore, the integration of planning tasks is essential and complements research focussed on optimising isolated processes.

2.1. Value of integrated planning

The primary incentive for implementing integrated planning is its potential value compared to separated planning approaches. Despite the promise that integrated planning holds for improving efficiency and coherence across various systems, there is a lack of empirical studies comparing the performance of integrated and separated planning algorithms, with most of the existing literature providing only qualitative discussions. A limited number of articles categorise and discuss integrated production and distribution planning (Fahimnia et al. 2013; Kumar et al. 2020; Moons et al. 2017) and procurement and production integration (Utama et al. 2022).

These reviews collectively agree on the current state of the art, from which three key findings can be derived. Firstly, mixed-integer linear programming (MILP) is more widely used than linear programming and simulation-based modelling (Fahimnia et al. 2013; Kumar et al. 2020). Despite their relatively straightforward formulation, MILPs are challenging to solve, and exact methods are only applicable to simple problems. Studies involving more complex environments generally use heuristics or metaheuristics (Kumar et al. 2020; Moons et al. 2017). Secondly, most models assume simplified problems with deterministic parameters (Fahimnia et al. 2013; Hrabec, Magnus Hvattum, and Hoff 2022; Kumar et al. 2020; Utama et al. 2022). Thirdly, tactical and operational planning currently predominates (Kumar et al. 2020), whereas strategic and tactical planning were more common in the past. This shift may indicate a trend towards more computationally intensive operational decision-making.

We were unable to identify any studies that directly compare integrated and separated planning in procurement and production. Nonetheless, there is limited evidence on the value of production-distribution integration within the existing literature. The study by Hrabec, Magnus Hvattum, and Hoff (2022) represents a notable

effort in this area to provide empirical evidence. They conducted a meta-analysis examining the benefits of integrating production, distribution, and inventory decisions. This study identified eleven case studies comparing integrated and sequential planning and fitted a random effects model to them. The model indicated expected cost savings from integration of 11.08% with a 95% confidence interval of [6.58%, 15.58%]. Similarly, Moons et al. (2017) estimated that integration can lead to savings of 5% to 20%. Bilgen and Günther (2010) solved a block planning problem in the fast-moving consumer goods industry and found that sequential planning leads to 10% to 15% higher costs than integrated planning. However, it remains difficult to identify general causal relationships between input parameters and cost savings due to the limited sample size of available literature. Instead, Hrabec, Magnus Hvattum, and Hoff (2022) synthesised claims from their literature sample, concluding that most studies agree on the increase in cost savings with higher system flexibility (e.g. number of products, customers, or jobs) and with higher distribution costs, plant capacity, or the ratio of fixed to variable transport costs. Krajewski and Wei (2001) examined how environmental factors influence the effectiveness of production schedule integration, finding that while integrated scheduling can lead to cost savings across the supply chain, some firms may incur higher costs compared to independent scheduling scenarios. Particularly, environments with high inventory holding costs and long supplier lead times may not benefit from schedule integration. The effectiveness of forecasts is also identified as a critical factor in realising the benefits of integrated scheduling.

2.2. Related case studies

So far, most research has focussed on integrated production-distribution planning (Fahimnia et al. 2013; Hrabec, Magnus Hvattum, and Hoff 2022; Kumar et al. 2020; Moons et al. 2017), while contributions to the integration of procurement and production planning remain limited (Utama et al. 2022). Therefore, this review focuses on case studies that propose integrated models in this area. Our focus is specifically on articles published after 2015, as older studies quickly become outdated due to rapid advances in algorithms and computational capabilities. We have analysed the case studies in terms of solution approaches, the use of real data, and the application of rolling horizon planning, decision levels and stochastic parameters. Table 1 classifies the identified studies and compares them with our work. As the classification of decision levels is a matter of definition, we follow the definition in the footnote of the table and also consult the length of the planning horizon mentioned in the articles.

All articles focus on production as the core process of supply chains and extend the problem to include decisions related to procurement, distribution, or both. Half of the case studies present models that return operational plans for at least one stage. Three articles specifically integrated operational procurement and production planning, similar to the problem addressed in this article (Gu, Gu, and Gu 2015; Shakhsi-Niaei and Sajadian 2021; Thevenin, Zufferey, and Glardon 2017). Gu, Gu, and Gu (2015) extended a flow shop scheduling problem to include material pickup and finished goods delivery, where pickup and delivery decisions determine

Table 1. Overview of related case studies ordered by year of publication.

Reference	Solution approach[a]	Real data	Rolling horizon	Stages (Decision level)[b]			Stochastics
				Proc.	Prod.	Dist.	
Our model	*GA, MILP, NN*	*x*	*x*	*O*	*O*	-	*Demand*
Cao and Wang (2022)	GA	x	–	T	T	T	–
Farghadani-Chaharsooghi et al. (2022)	GA, MILP	x	–	–	T	O	Demand, Lead time
Benbouja et al. (2021)	MILP	–	–	T	T	T	Demand
Alavidoost, Jafarnejad, and Babazadeh (2021)	GA	–	–	T	T	T	Demand, Cost
Shakhsi-Niaei and Sajadian (2021)	MILP	x	–	O	O	–	–
Darvish and Coelho (2018)	MILP, Matheuristic	x	–	–	T	T	–
Thevenin, Zufferey, and Glardon (2017)	Greedy, TS, VNS, MILP	x	x	O	O	O	Demand
Sawik (2016)	MILP	x	–	T	T	T	Disruptions
Gu, Gu, and Gu (2015)	GA	–	–	O	O	O	–
Sarrafha et al. (2015)	BBO, GA, Johnson	–	–	T	O	T	Demand
Shirvani and Shadrokh (2013)	MILP	–	–	T	O	T	–
Krajewski and Wei (2001)	–	–	x	T	T	–	Demand

[a]BBO: Bigeography-based optimisation; GA: Genetic algorithm; MILP: Mixed-integer linear programming; Neural network; TS: Tabu search; VNS: Variable neighbourhood search[b]O: Operational planning involves creating detailed, short-term plans for procurement, production, and distribution, specifying exact actions, schedules, quantities, and resources needed.; T: Tactical planning involves setting short- to medium-term targets for order quantities, production quantities, transport quantities, and safety stock levels, typically on a weekly or monthly basis.

the timing of material transport between facilities and the plant using vehicles with limited capacity. Thevenin, Zufferey, and Glardon (2017) solved a scheduling problem to derive procurement and distribution plans, involving multiple products that must be scheduled on non-identical parallel machines with sequence-dependent setup times. Based on the solution, they calculate the costs associated with the timely procurement of materials and the distribution of finished goods to customers. Shakhsi-Niaei and Sajadian (2021) follow a similar approach. They first solved the scheduling problem and then derived a procurement plan from the solution, choosing from three ordering strategies. Their scheduling problem stands out as it is the only study to consider a job shop, recognised as the most complex machine environment. Other operational production planning models focus on parallel identical machines (Shirvani and Shadrokh 2013) or flow shops (Gu, Gu, and Gu 2015; Sarrafha et al. 2015). Some studies also consider special features such as lead times (Shakhsi-Niaei and Sajadian 2021). Although multi-machine environments and sophisticated job characteristics are state of the art for isolated scheduling optimisation, integrated models still predominantly address simpler scheduling problems. Consistent with the bias towards production-distribution integration in the literature review articles, distribution planning is highly represented in our literature sample. This is particularly surprising at the operational level, given that companies often outsource distribution to third party logistics providers, limiting their influence on specific routing decisions. One possible explanation is the popularity of the vehicle routing problem in academia, which typically embodies the task of distribution planning. Although the literature sample does not show a significant trend towards lower decision levels (see Table 1), the field of integrated planning has progressed towards addressing more complex and realistic problems. All studies employ analytical models to describe their optimisation problems, with MILP solvers and GAs being the dominant solution approaches. It is also common to combine the advantages of both approaches: several authors used MILP solvers on small instances to validate their metaheuristics, which are then applied to larger instances (Farghadani-Chaharsooghi et al. 2022; Thevenin, Zufferey, and Glardon 2017). Meanwhile, some authors opt for pure MILP-based optimisation approaches, despite the limitations of handling relatively small instances and enduring long computation times (Sawik 2016; Shakhsi-Niaei and Sajadian 2021; Shirvani and Shadrokh 2013).

About half of the articles incorporate randomness or uncertainty through stochastic parameters. Stochastic parameters provide a more realistic representation of real-world supply chains, which are characterised by inherent uncertainty in factors such as demand, lead time, and cost. Ignoring this variability can lead to overly simplistic or inaccurate models. In the literature, stochastic demand is typically represented by distribution functions (Krajewski and Wei 2001; Thevenin, Zufferey, and Glardon 2017), scenarios (Farghadani-Chaharsooghi et al. 2022), or fuzzy methods (Alavidoost, Jafarnejad, and Babazadeh 2021) is common in the literature. Other stochastic factors are still rare. An exception is the study by Sawik (2016) which considered integrated planning under stochastic disruptions, modelled using a multinomial discrete distribution. One approach to dealing with stochastic environments is rolling horizon planning which is common practice in industry but has received less attention in academia (Sahin, Narayanan, and Robinson 2013). Although the inclusion of rolling horizons in integrated planning is rare, it is a promising method for reacting to changes in stochastic environments through re-planning. Only two models in our literature sample consider rolling horizons (see Table 1). In the study by Krajewski and Wei (2001), the manufacturer updates its material commitments in each period and communicates them to the suppliers so that they can adjust their production schedules. Thevenin, Zufferey, and Glardon (2017) model the rolling horizon through a frozen period constraint for managerial reasons. While the sequence and size of the production blocks are fixed, the assignment of jobs to these blocks remains flexible as long as the product type matches. This model allows for adjustments to the production schedule after the initial fixed period to accommodate changes in demand or production capacity while maintaining short-term stability.

2.3. Research gap and contribution

Although some research addresses the integration of specific planning problems, there is a notable lack of empirical studies that quantitatively compare the performance of integrated versus separated planning approaches. Specifically, the value of integrating procurement and production decisions has not yet been quantitatively examined in the literature, despite the presence of some qualitative discussions. Due to this lack of evidence, Hrabec, Magnus Hvattum, and Hoff (2022) highlighted the need for additional comparative studies that empirically evaluate the benefits of integrated planning based on real-world cases. This gap is particularly evident in operational planning, where the complexity and high computational efficiency requirements make empirical evaluation difficult (Thevenin, Zufferey, and Glardon 2017). Although three articles consider detailed

procurement-production planning problems, these models are different to ours (Gu, Gu, and Gu 2015; Shakhsi-Niaei and Sajadian 2021; Thevenin, Zufferey, and Glardon 2017). These studies mainly focus on optimising the production stage, with procurement treated as a dependent extension to evaluate the total cost of scheduling decisions. As a result, there is no flexibility in choosing suppliers, order quantities, or order dates. Furthermore, rolling horizon planning remains a research gap in integrated planning that has not received sufficient attention from academia (Fahimnia et al. 2013).

Our study aims to bridge the gap between the reality of industrial problems and the problems addressed in OR. Unlike previous studies, we model both stages as full optimisation problems and link them through an integrated objective function and additional constraints. We propose and evaluate several integrated and sequential solution approaches to provide empirical evidence on the value of integration. The empirical evaluation involves testing a large number of scenarios, including rolling horizon instances to account for stochastic demand. This testing framework ensures that our findings are robust and applicable to different real-world settings. Our problem formulation and solution approaches are formulated in a general way, making the concepts and methods applicable to a wide range of industrial contexts. To the best of our knowledge, our work is the first to apply a GA combined with a neural network for integrated planning. This novel method generates effective decisions in a limited time frame, making it well suited for operational planning. In addition, our study is among the first to empirically compare integrated and sequential planning approaches for procurement and production under an operational rolling horizon policy.

3. Problem description

The problem considered is an integrated order allocation and scheduling problem based on a real case study which we describe in detail in Section 5. The aim is to find procurement and production schedules that minimise total costs while maintaining a sufficient level of performance regarding on-time delivery of customer orders. The manufacturer employs a make-to-order manufacturing strategy, linking manufacturing jobs directly to customer demands. We begin by formulating the isolated order allocation (see Section 3.1) and scheduling problems (see Section 3.2). Subsequently, we introduce the integrated problem by linking the two isolated ones (see Section 3.3). The model formulations do not incorporate dynamic or stochastic parameters inherently because we employ an event-based re-planning, rolling horizon approach (see Section 4). Stochastic elements are isolated in an external simulation module that generates random disruptions which trigger the need for re-planning.

3.1. Order allocation problem

The OAP determines the optimal quantities of raw materials to order from various suppliers over a given planning horizon, along with when to place these orders and how to manage inventory levels. The problem is only concerned with special, job-specific materials, the management of generic supplies is not part of the optimisation model. The OAP is crucial for maintaining the balance between minimising costs and ensuring that inventory levels are sufficient to meet demand without incurring shortages or excessive surplus. Our specific problem allows for regular and emergency orders. Regular orders require a lead time until they arrive in stock, while emergency orders are directly available for the price of higher variable costs. We model the problem under the following assumptions:

- The procurement environment is static and deterministic. Demands, lead times and costs are known and fixed. Suppliers are reliable and will deliver orders as agreed upon in terms of quantity and time.
- Demand must always be met and backorders are not allowed. If a demand cannot be satisfied by regular orders, emergency orders must compensate the shortage.
- Each material can be sourced from multiple suppliers simultaneously. The multiple sourcing strategy allows to mitigate risks by diversifying the supplier base.
- The inventory holding capacity of each material is unlimited.

We formulate the OAP as a MILP using the notation presented in Table 2. The objective is to minimise the total cost summed over all periods of the planning horizon. Total cost include fixed ordering costs, regular and variable ordering costs for each type of material from each supplier and inventory holding costs for each material (see Equation (1)).

The inventory balance Equation (1a) calculates the inventory at the end of the period based on the inventory left from the previous period, the sum of regular and emergency orders arriving in the current period and the demand of the current period. We set $q_{t-L_{s,r},s,r} = 0$ for $L_{s,r} \geq t$. Constraint (??) ensures that fixed costs occur whenever a regular order is placed with a supplier. Constraint (1b) ensures that orders (both regular and emergency) are only placed for materials that are actually produced by the supplier s. The remaining constraints are domain constraints for the decision variables.

Table 2. Notation for indices, parameters and variables of the order allocation problem.

Index	Description
$t \in \mathcal{T} = \{1, \ldots, T\}$	Planning periods
$s \in \mathcal{S} = \{1, \ldots, S\}$	Suppliers
$r \in \mathcal{R} = \{1, \ldots, R\}$	Raw materials
Parameter	**Description**
$J_{t,r}$	Demand for material r in period t
$V_{s,r}$	Variable cost for one unit of material r from supplier s
$E_{s,r}$	Emergency cost for one unit of material r from supplier s
F	Fixed order cost
H_r	Cost of holding inventory for material r
$L_{s,r}$	Lead time for material r at supplier s
$L_{\max}$	Maximum lead time, $L_{\max} = \max_{s,r} L_{s,r}$
$O_{t,s,r}$	Incoming units of material r from supplier s arriving in period t (from previously placed orders)
$Y_{s,r}$	Binary parameter, 1 if supplier s produces material r
$I_{0,r}$	Initial inventory level of material r
M_1	A large number
Decision variable	**Description**
$q_{t,s,r}$	Regular order quantity of material r from supplier s ordered in period t
$e_{t,s,r}$	Emergency order quantity of material r from supplier s in period t
$I_{t,r}$	Inventory of material r at the end of period t
$f_{t,s}$	Binary variable, 1 if a regular order is placed with supplier s in period t

Constraint (1e) prevents backorders.

$$\min_{q,e,f,I} \quad \sum_{t\in\mathcal{T}} \left(F \sum_{s\in\mathcal{S}} f_{t,s} + \sum_{s\in\mathcal{S}, r\in\mathcal{R}} (V_{s,r} q_{t,s,r} + E_{s,r} e_{t,s,r}) + \sum_{r\in\mathcal{R}} H_r I_{t,r} \right) \tag{1}$$

s.t.

$$I_{t,r} = I_{t-1,r} + \sum_{s\in\mathcal{S}} \left(q_{t-L_{s,r},s,r} + O_{t,s,r} + e_{t,s,r} \right) - J_{t,r} \quad \forall\, t, r \tag{1a}$$

$$\sum_{r\in\mathcal{R}} q_{t,s,r} \leq M_1 f_{t,s} \quad \forall\, t, s \tag{1b}$$

$$e_{t,s,r} + q_{t,s,r} \leq M_1 Y_{s,r} \quad \forall\, t, s, r \tag{1c}$$

$$q_{t,s,r}, e_{t,s,r} \in \mathbb{N}_0 \quad \forall\, t, s, r \tag{1d}$$

$$f_{t,s} \in \{0, 1\} \quad \forall\, t, s \tag{1e}$$

$$I_{t,r} \geq 0 \quad \forall\, t, r \tag{1f}$$

3.2. Scheduling problem

The scheduling problem decides when to process which job on which machine. We address a machine environment known as a flexible or hybrid flow shop, characterised by multiple sequential stages, each equipped with several homogeneous parallel machines. The environment is complicated further by sequence-dependent setup times, which vary according to the product families of the jobs, and a specific setup constraint that prohibits identical setup configurations on parallel machines simultaneously. We model the scheduling problem under the following assumptions:

- The machine environment is static and deterministic. Processing times, setup times and due dates are known and fixed. Machines operate without failures or maintenance breaks.
- There is only one setup device available per job family, implying that different jobs within the same family cannot be processed simultaneously.
- The machines are homogeneous, implying that any machine at a given stage can process any job designated for that stage.

The MILP formulation is an adaptation of the work by Jungwattanakit et al. (2008). We use the notation presented in Table 3 to formulate the mathematical model. The objective is to minimise the total tardiness cost across all jobs (see Equation (2)).

Constraint (2a) guarantees that the model recognises running jobs by scheduling them first. The constraint is only implemented for those stages and machines, on which a job is running. Constraints (2b) and (2c) ensure that each job is processed exactly once per stage by specifying that each job has exactly one predecessor and one successor. Constraints (2d) and (2e) ensure that each machine initiates and concludes the sequence with exactly one job. Constraint (2f) prohibits any job from preceding itself. Constraint (2g) ensures that each job is only scheduled on one machine per stage. Constraints (2h) and (2i) define lower bounds on the job completion time considering (initial) setup, previous job completion time (if any) and processing time. Constraint (2j) ensures that the completion time of a job is no earlier than its completion time at the previous stage plus any setup and processing times incurred at the current stage. The tardiness of a job is determined by constraint (2k), based on the difference between its completion time at the final stage and its due date. Constraint (2l) is an auxiliary equation calculating the start time for each job at each stage by subtracting processing and setup time from its completion time. The start time is relevant for the constraint. Constraint (2m) addresses a specific setup limitation that prevents the simultaneous scheduling of jobs from the same product family due to restricted setup resources. In our case study, this is only

Table 3. Notation for indices, parameters and variables of the scheduling and integrated problem.

Scheduling problem	
Index	Description
$k \in \mathcal{K} = \{1, \dots, K\}$	Stages in the production system
$i \in \mathcal{M}^k = \{1, \dots, M^k\}$	Machines available at stage k
$j, l \in \mathcal{N}^k$	Jobs to be processed at stage k
α, ω	Placeholder indices representing the start and end of the job sequence on a machine
$\mathcal{U} = \{U_1, U_2, \dots\}$	Product families. Each job belongs to exactly one family.
Parameter	Description
M^k	Number of parallel machines at stage k
D_j	Due date of job j
$Q^k_{j,l}$	Setup time required at stage k for job l if it follows job j
$A^k_{j,l}$	Initial setup time at stage k for job l if it is the first job
P^k_j	Processing time for job j at stage k
C_j	Tardiness cost per time unit for job j
M_2	A large constant
B^k_i	Job currently running on machine i at stage k (if any)
Decision variable	Description
$x^k_{i,j,l}$	Binary variable, 1 if job j is scheduled directly before job l on machine i at stage k
c^k_j	Completion time of job j at stage k
a_j	Tardiness of job j
$g^k_{j,l}$	Binary variable, 1 if job j is scheduled before (not necessarily directly before) job l at stage k. This variable is only created for stage $k = 1$ and for job pairs j, l of the same product family.
b^k_j	Start time of job j at stage k
Integrated problem	
Parameter	Description
G	Time conversion factor (number of scheduling time units in one procurement period)
$Z_{j,r}$	Required amount of raw material r for job j
Decision variable	Description
$J_{t,r}$	Demand of material r in period t
$p_{j,t}$	Binary variable, 1 if the processing of job j starts in period t (only for jobs yet to be scheduled but not yet running in stage 1, i.e. for $j \in N_1 := \mathcal{N}^1 \setminus \{B^1_i, i \in M^1\}$)

required on stage $k = 1$. The constraint is only formulated for job pairs j, l that belong to the same product family. Unambiguous precedence relations are ensured by constraint (2n). Lastly, constraints (2o) and (2p) define the domains of the decision variables.

$$\min_{x,c,a,g,b} \quad \sum_{j \in \mathcal{N}^K} C_j a_j \tag{2}$$

s.t

$$x^k_{i,\alpha,B^k_i} = 1 \quad \forall\, k, i \tag{2a}$$

$$\sum_{i \in \mathcal{M}^k} \sum_{j \in \mathcal{N}^k \cup \alpha} x^k_{i,j,l} = 1 \quad \forall\, k, l \tag{2b}$$

$$\sum_{i \in \mathcal{M}^k} \sum_{l \in \mathcal{N}^k \cup \omega} x^k_{i,j,l} = 1 \quad \forall\, k, j \tag{2c}$$

$$\sum_{l \in \mathcal{N}^k \cup \omega} x^k_{i,\alpha,l} = 1 \quad \forall\, k, i \tag{2d}$$

$$\sum_{j \in \mathcal{N}^k \cup \alpha} x^k_{i,j,\omega} = 1 \quad \forall\, k, i \tag{2e}$$

$$x^k_{i,j,j} = 0 \quad \forall\, k, i, j \tag{2f}$$

$$\sum_{j \in \mathcal{N}^k \cup \alpha} x^k_{i,j,l} = \sum_{j \in \mathcal{N}^k \cup \omega} x^k_{i,l,j} \quad \forall\, k, i, l \tag{2g}$$

$$c^k_l \geq \sum_{i \in \mathcal{M}^k} x^k_{i,\alpha,l} A^k_{i,l} + P^k_l \quad \forall\, k, l \tag{2h}$$

$$c^k_l \geq c^k_j + Q^k_{j,l} + P^k_l + \left(\sum_{i \in \mathcal{M}^k} x^k_{i,j,l} - 1 \right) M_2 \quad \forall\, k, j, l \in \mathcal{N}^k \setminus \{j\} \tag{2i}$$

$$c^k_l \geq c^{(k-1)}_l + \sum_{i \in \mathcal{M}^k} \left(A^k_{i,l} x^k_{i,\alpha,l} + \sum_{j \in \mathcal{N}^k} Q^k_{j,l} x^k_{i,j,l} \right) + P^k_l \quad k \geq 2, l \in \mathcal{N}^{k-1} \tag{2j}$$

$$a_j \geq c^K_j - D_j j \in \mathcal{N}^K \tag{2k}$$

$$b^k_l = c^k_l - P^k_l - \sum_{i \in \mathcal{M}^k} \left(A^k_{i,l} x^k_{i,\alpha,l} + \sum_{j \in \mathcal{N}^k} Q^k_{j,l} x^k_{i,j,l} \right) \quad \forall\, k, l \tag{2l}$$

$$c^k_j \leq b^k_l + (1 - g^k_{j,l}) M_2 \quad k = 1, U \in \mathcal{U}, j, l \in U \cap \mathcal{N}^1 \tag{2m}$$

$$g^k_{j,l} = 1 - g^k_{l,j} \quad k = 1, U \in \mathcal{U}, j, l \in U \cap \mathcal{N}^1 \tag{2n}$$

$$g^k_{j,l}, x^k_{i,j,l} \in \{0, 1\} \quad \forall\, k, i, j, l \tag{2o}$$

$$c^k_j, a_j, b^k_j \geq 0 \quad \forall\, k, j \tag{2p}$$

3.3. Integrated problem

The integration of the order allocation and scheduling problems requires a few new components to connect both models. All assumptions, parameters, decision variables and constraints from Sections 3.1 and 3.2 remain valid with one exception. The parameter $J_{t,r}$ from the OAP is internalised into a decision variable, as the demand is now imposed by the material requirements of the scheduling plan. For the computation of the material requirements, we rely on a new parameter $Z_{j,r}$ that specifies the raw material requirements of each job, and on a new binary variable $p_{j,t}$ that indicates the OAP time

period in which a job is started. Additionally, the integrated model incorporates a time conversion factor G to make the different time units of the OAP (usually days) and scheduling problem (usually minutes) compatible.

The new objective function (see Equation (3)) evaluates the quality of an integrated procurement and production plan. The objective value of an integrated plan is the sum of the procurement and production plan objectives. It replaces the objectives of the separated models in the optimisation problem. We use the notation presented in Table 3 to formulate the mathematical model.

Constraint (3a) ensures that the material demands for each OAP period match the sum of material requirements for all jobs scheduled in that period, based on their individual material requirements ($\alpha_{j,r}$). The constraints (3b), (3c) and (3d) identify the correct, unambiguous OAP period a job starts being processed. ST marks the beginning of the setup operations. Since we require raw materials only once the actual processing begins, we add the setup time.

$$\min_{\substack{x,c,a,g,b,\\ p,J,q,e,f,I}} \quad \sum_{j\in\mathcal{N}^K} C_j a_j + \sum_{t\in\mathcal{T}}\Bigg(F\sum_{s\in\mathcal{S}} f_{t,s} + \sum_{s\in\mathcal{S},r\in\mathcal{R}} (V_{s,r}q_{t,s,r} + E_{s,r}e_{t,s,r}) + \sum_{r\in\mathcal{R}} H_r I_{t,r}\Bigg) \tag{3}$$

s.t

$$J_{t,r} = \sum_{j\in N_1} Z_{j,r}p_{j,t} \quad \forall\, t,r \tag{3a}$$

$$G\sum_{t\in\mathcal{T}} t p_{l,t} \geq b_l^1 + \sum_{i\in\mathcal{M}^1}\left(A_{i,l}^1 x_{i,\alpha,l}^1 + \sum_{j\in\mathcal{N}^1} S_{j,l}^1 x_{i,j,l}^1\right) \quad l\in N_1 \tag{3b}$$

$$G\sum_{t\in\mathcal{T}} (t-1) p_{l,t} \leq b_l^1 + \sum_{i\in\mathcal{M}^1}\left(A_{i,l}^1 x_{i,\alpha,l}^1 + \sum_{j\in\mathcal{N}^1} S_{j,l}^1 x_{i,j,l}^1\right) \quad l\in N_1 \tag{3c}$$

$$\sum_{t\in\mathcal{T}} p_{l,t} = 1 \quad l\in N_1 \tag{3d}$$

4. Solution modules & solution approaches

Like most of the related literature, we face an optimisation problem that has not been addressed before. Even closely related work (e.g. Gu, Gu, and Gu 2015; Shakhsi-Niaei and Sajadian 2021; Thevenin, Zufferey, and Glardon 2017) differs so fundamentally in terms of scheduling and procurement environments that it is difficult to adapt them to our problem without substantial modifications. Among these, only the Variable Neighbourhood Search (VNS) algorithm proposed by Thevenin, Zufferey, and Glardon (2017) is partially suitable for adaptation to our context. Therefore, we integrate an adapted version of this algorithm with our OAP formulation to serve as a baseline for performance comparison. The modifications are given in Appendix 3.

A key contributions of our work is addressing the fundamental research question of how to efficiently integrate scheduling and procurement decisions within our problem setting. To answer this question, we propose three variants of a novel integrated solution approach. These variants differ with respect to the problem complexity for which they are suitable. To contextualise the performance of these three approaches, we compare them with an integrated random search. A second identified research gap is the lack of empirical studies investigating the benefits of integrated versus separated decision-making for operational OR problems. To this end, we analyse three separated approaches. Two of these methods are direct counterparts of two of the integrated methods and are therefore well suited for direct comparison. The third separated approach mimics the status quo of how the company makes scheduling and procurement decisions, allowing us to investigate the company-specific consequences of switching to our proposed approach. In the following two sections, we present all solution approaches applied in this work. As our solution approaches follow a modular design, the basic solution modules are described first (see Section 4.1), followed by an explanation of how these modules are combined to form the solution approaches.

4.1. Solution modules

Each of our solution approaches is a combination of three basic solution modules: MILP, GA, and LSTM-based neural networks. The MILP formulations were introduced in Section 3. The GA and LSTM networks are introduced in this subsection, while the necessary modifications of the baseline approach proposed by Thevenin, Zufferey, and Glardon (2017) are presented in Appendix 3.

4.1.1. Genetic algorithm

Given that the scheduling problem is NP-hard (Jungwattanakit et al. 2008), the integrated procurement-production problem is, too. For problems of this complexity, there are no known methods that generate an

optimal solution in polynomial time. Consequently, OR researchers often turn to metaheuristics (Griffis, Bell, and Closs 2012). In the related literature, nature-inspired metaheuristics such as evolutionary and swarm intelligence algorithms (Rajwar, Deep, and Das 2023) are particularly popular (see Section 2.2). While other metaheuristics, especially from these two areas, are also applicable, we choose to use a GA (Goldberg 1989; Holland 1992) as the central component of our solution approaches. GAs are the most common solution method in the related literature (see Table 1) and one of the most popular metaheuristics in OR (Griffis, Bell, and Closs 2012) and heuristic optimisation in general (Rajwar, Deep, and Das 2023). One of the main strengths of GAs is their ability to efficiently explore large and complex search spaces, making them suitable for NP-hard problems. However, GAs can be computationally intensive due to the evaluation of a large number of candidate solutions. Alternative metaheuristics such as VNS and Tabu Search offer different approaches to optimisation. While these methods can be less computationally intensive and simpler to implement in certain contexts, they may be more prone to getting trapped in local optima compared to GAs due to their reliance on local search strategies. In this work, we choose GAs for their ability to deal with the complex and large solution space of this integrated problem. In addition, we reduce the computational effort This approach combines the speed of simpler metaheuristics with the capabilities of the GA. In the following section, we only discuss the problem-specific features of our GA implementation. For a general introduction to GAs, the reader is referred to Eiben and Smith (2015). Figure 1 presents the schema of our GA approach.

For a GA to produce a good solution, it may need to run for a large number of generations. As we aim to make short-term, operational decisions, large runtimes are to be avoided, necessitating computational efficiency. Contrary to other authors (Sarrafha et al. 2015), we therefore decide against complex individuals that represent a solution to the integrated problem but concentrate on a simplified version of the scheduling subproblem. This allows for a compact solution representation and straightforward, traditional mutation and crossover operators that would otherwise require non-trivial modifications and time-consuming repair operations. Consequently, our individuals are represented by $K+1$ vectors $v_i, i = 1, \ldots, K+1$. A visualisation, explained in more detail below, is provided in Figure 2. The first K vectors, the job sequence vectors, are permutations of the list of job IDs to be scheduled, the ith vector defines the sequence in which the jobs are scheduled in the ith stage. The assignment to individual machines is automated such as to minimise idle time (see Figure 5). We emphasise that these vectors do not have to be of the same length, since jobs to be processed in a later stage may have been completed in an earlier stage, i.e. $\dim(v_i) \geq \dim(v_j), i > j$. In addition, to reduce procurement costs by avoiding emergency orders, it can be beneficial to postpone the start of processing a job in stage one. We

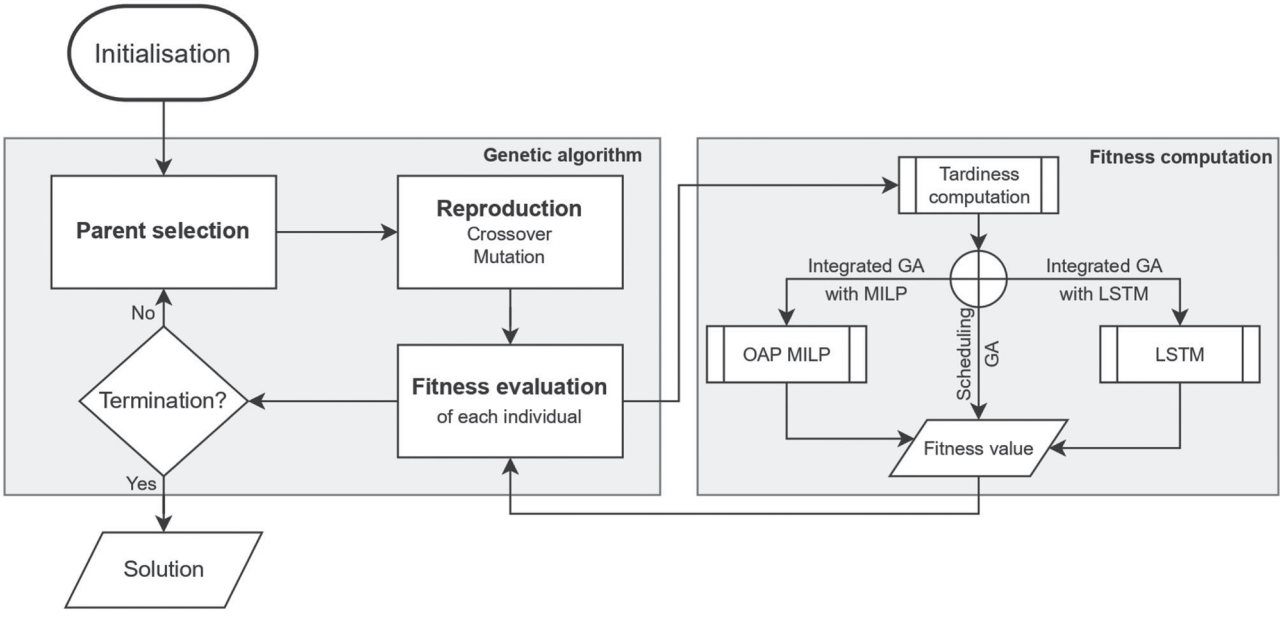


Figure 1. Structure of the GA. The three variants (scheduling GA, integrated GA with OAP MILP-based fitness, and integrated GA with LSTM-based fitness) rely on the same basis, i.e. they share the same individual representation, selection, crossover and mutation operators. They only differ in the fitness computation.

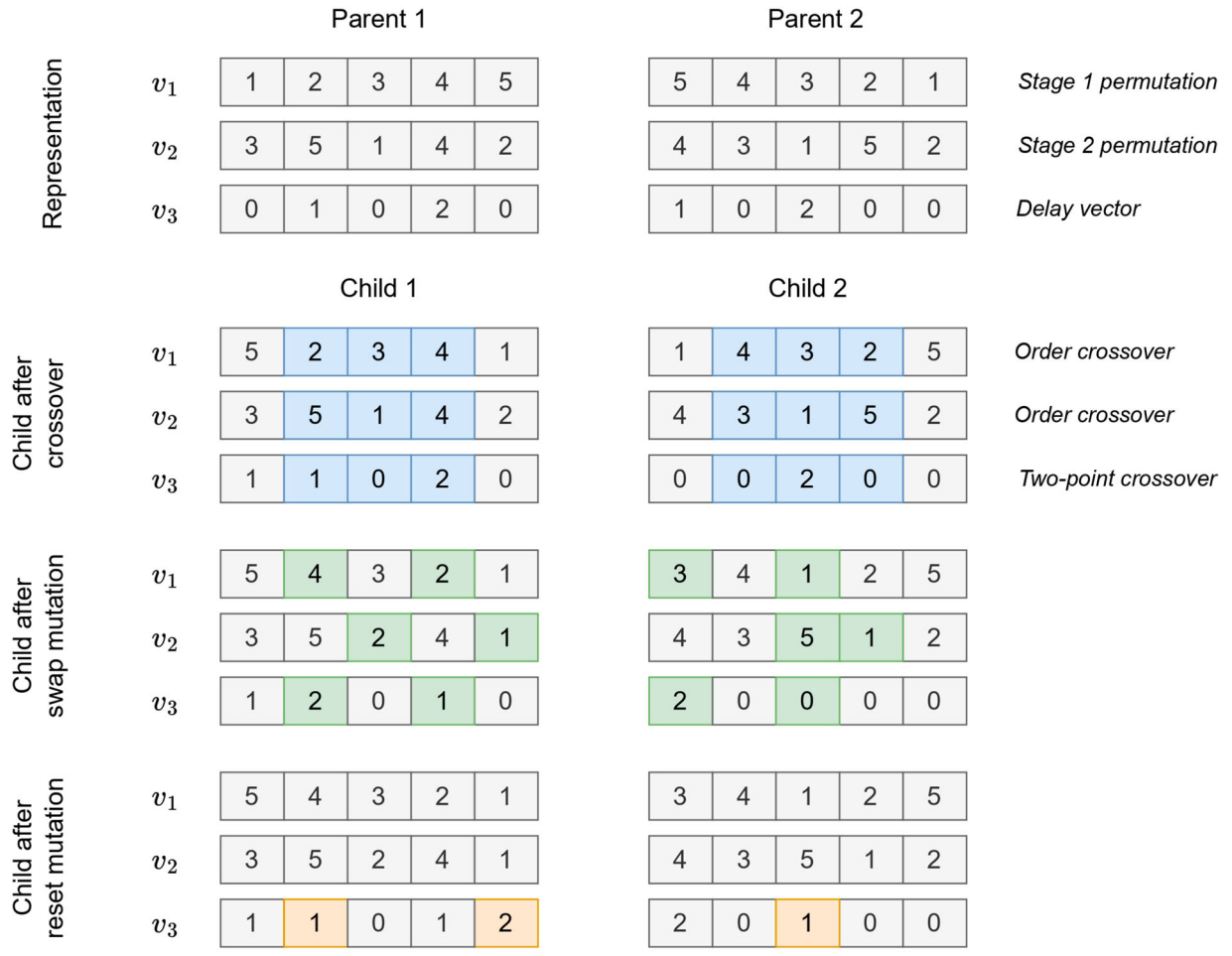


Figure 2. Example of crossover and mutation operators for a toy example with two production stages ($K = 2$), five jobs and $max\_delay = 2$.

therefore have an additional vector, the delay vector v_{K+1}, which has the same length as v_1. An entry of n, where $n \in \{0, 1, \ldots, max\_delay\}$, at the ith position of v_{K+1} pushes the corresponding job $v_{1,i}$ to the start of the nth next day.

The objective value of the scheduling problem, the weighted tardiness, can be computed analytically for any individual in negligible time. This is sufficient to obtain the individual's fitness when using the GA to solve the scheduling problem only. We refer to this GA variant as the *scheduling GA* (see Figure 1).

However, we also want to use the GA to solve the integrated problem. To this aim, it needs to (a) rank the individuals in terms of their integrated fitness and (b) return a solution to the integrated problem (even though the internal representation of the individuals provides a solution to the scheduling problem only). These issues can be solved in two ways. Each scheduling solution defines specific material requirements, which are the demand parameters for the OAP. We can therefore solve the MILP formulation of the OAP corresponding to the scheduling solution, which returns both an optimal procurement plan and its cost. We can then add the procurement cost to the weighted tardiness to obtain the individual's integrated fitness value (see Figure 1). However, computational efficiency is also important regarding fitness evaluation (Goldberg 1989), and although the OAP is simpler than the scheduling problem, solving the OAP for each individual is costly. Furthermore, to rank the individuals, we do not need to compute a detailed procurement plan, but only the procurement cost. Therefore, we propose an alternative approach that uses a pre-trained LSTM-based neural network to estimate the optimal procurement cost in real-time. The neural network is described in Section 4.1.2. Regardless of the method employed to obtain the procurement cost, this extended GA variant used to solve the integrated problem is referred to as the *integrated GA* (see Figure 1).

To implement the GA, we need to specify the selection, crossover, and mutation operators (see Figure 1). To select the parents of the next generation, we use deterministic tournament selection (Miller and Goldberg 1995). Tournament selection does not require an

exact fitness value but only a relative ranking, which is particularly appropriate when we use the LSTM network to approximate procurement costs, is fast to apply, and the tournament size allows direct control of selection pressure. The selected parent pool is divided into random pairs of two, and each pair of parents produces two children. With a crossover probability of $p_c \in [0, 1]$, the children are the crossover results of their parents, while with probability $1 - p_c$ they are unmodified copies. Our simplistic individuals allow us to apply the traditional and well-known order crossover operator (Davis 1991) to the job sequence vectors, and the popular n-point crossover (Eiben and Smith 2015) with $n = 2$ to the delay vector. Then, each child is mutated with mutation probability $p_m \in [0, 1]$ or added to the next generation unmodified with probability $1 - p_m$. We apply swap mutation to the job sequence vectors, and reset mutation to the delay vector (Eiben and Smith 2015). Figure 2 demonstrates crossover and mutation operators using a toy example of five jobs.

Finally, to avoid a decrease in solution quality, we apply elitism (Russell and Norvig 2022) by carrying over the best individuals from one generation to the next unaltered. The interested reader finds a more detailed description of our GA including pseudocode in Section 1 of the appendix.

4.1.2. Long short-term memory neural networks

Recurrent neural networks (RNNs) are a type of artificial neural network specifically designed to process sequence data, such as natural language or time series (Chollet 2018). Unlike other types of neural networks, RNNs handle inter-temporal dependencies explicitly, efficiently share weights across input vectors and can deal with input sequences of varying length. Due to the vanishing gradient problem, simple RNNs still struggle to learn long-term dependencies (Bengio, Simard, and Frasconi 1994). Long-short term memory (Hochreiter and Schmidhuber 1997) is a technique employed in RNNs that has proven effective in dealing with this issue (Chollet 2018).

We employ multi-input, LSTM-based networks as a subroutine in our GA to predict the optimal procurement cost associated with a scheduling solution (see Section 4.1.1). This allows us to bypass the OAP-MILP, saving valuable computation time that can be allocated to the GA's search effort instead. Figure 3 shows the structure of our networks. Procurement cost parameters (fixed, variable, emergency, and holding cost matrices) are flattened and concatenated into a single column vector before being processed by a feed-forward (in `keras`: 'Dense') module. Material information (matrices of material requirements, incoming orders and starting inventory) are joined to a matrix of dimensions $T \times 2R$, where the left $T \times R$ matrix is J, the right $T \times R$ matrix represents current inventory and incoming orders over time. Material information is processed by an LSTM module. The resulting cost and material vectors are concatenated, further processed by a joint feed-forward module, and procurement costs are estimated. A crucial detail of our networks is the separate processing of procurement and material information, for two reasons. First, these inputs are structurally different, as cost parameters are constant matrices of fixed dimensions while material information has a time component and varies in dimensions depending on the scheduling solution. Therefore, the latter requires an RNN architecture. Second, the separate but explicit processing of cost parameters allows us to use the same network for different cost scenarios. In fact, we use only one network for all 72 cost scenarios on which we evaluate our approach (see Section 5). This is of practical importance because, once trained, the network is robust to supplier-induced changes in the cost structure.

4.1.3. Tabu search and reactive variable neighbourhood search

To contextualise the performance of our proposed approach, we adopt and, where needed, adapt the method put forward by Thevenin, Zufferey, and Glardon (2017) as a baseline. In their article, the authors integrate a

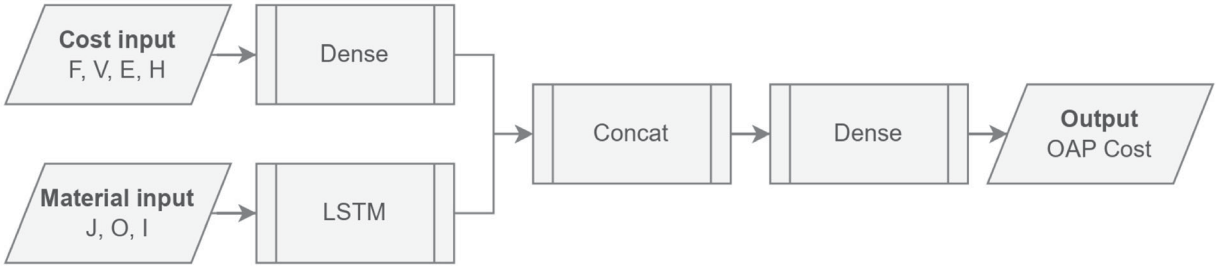


Figure 3. Architecture of the neural network approximating the optimal procurement cost of a scheduling solution. Cost and material information are processed separately. LSTM represents an LSTM-cell, Dense a feed-forward cell, and Concat a vector concatenation layer.

tabu search heuristic into a reactive variable neighbourhood search algorithm. The details of this approach are presented in Section 3 of the Appendix.

4.2. Solution approaches

Each of our ten solution approaches is a combination of the solution modules described in Sections 3 and 4.1. The following subsections detail how the modules are combined to create the solution approaches. Figure 4 provides a graphical overview to support our explanations.

4.2.1. Baseline ordering and scheduling strategy (StatusQuo)

The StatusQuo approach mimics the company's current ordering and scheduling strategies, which are as follows. First, the company manually creates a schedule using the domain knowledge of its decision-makers. Next, these decision-makers formulate a procurement plan that meets the material requirements derived from the schedule. The current method therefore represents a separated decision-making approach. The primary objective is to minimise setup times while maintaining a low level of tardiness.

Since the current scheduling strategy is entirely manual, it cannot be fully represented as a heuristic. Therefore, we replicate the scheduling strategy as a custom variant of the Earliest Due Date (EDD) dispatching rule that takes batching into account. The process involves three main steps: First, jobs are grouped into batches based on their product families. In the company's production environment, which is characterised by long setup times when switching between product families (see Section 5.1), this ensures a low total setup time. Second, the batches are sorted in ascending order by the mean due date of the jobs within each batch, and the jobs within each batch are sorted by their due date. This achieves low levels of tardiness. Third, the optimal procurement plan is determined by solving the OAP MILP.

4.2.2. Separated mixed-integer linear program (SepMILP)

The SepMILP approach is the second of the four approaches that solve the scheduling and procurement problems separately and allows us to measure the benefits of integrated decisions by comparing them with integrated methods. SepMILP first computes the optimal solution to the scheduling problem by solving the scheduling MILP. This results in a tardiness-minimising schedule. SepMILP then computes the optimal procurement plan pertaining to the optimal scheduling solution by solving the order allocation MILP. As the scheduling problem is NP-hard, this approach is only feasible for small problem instances (see Figure 4(b)). SepMILP is the separated counterpart to IntMILP and the exact counterpart to SepGA, and is therefore best suited for comparisons with these two approaches.

4.2.3. Separated genetic algorithm (SepGA)

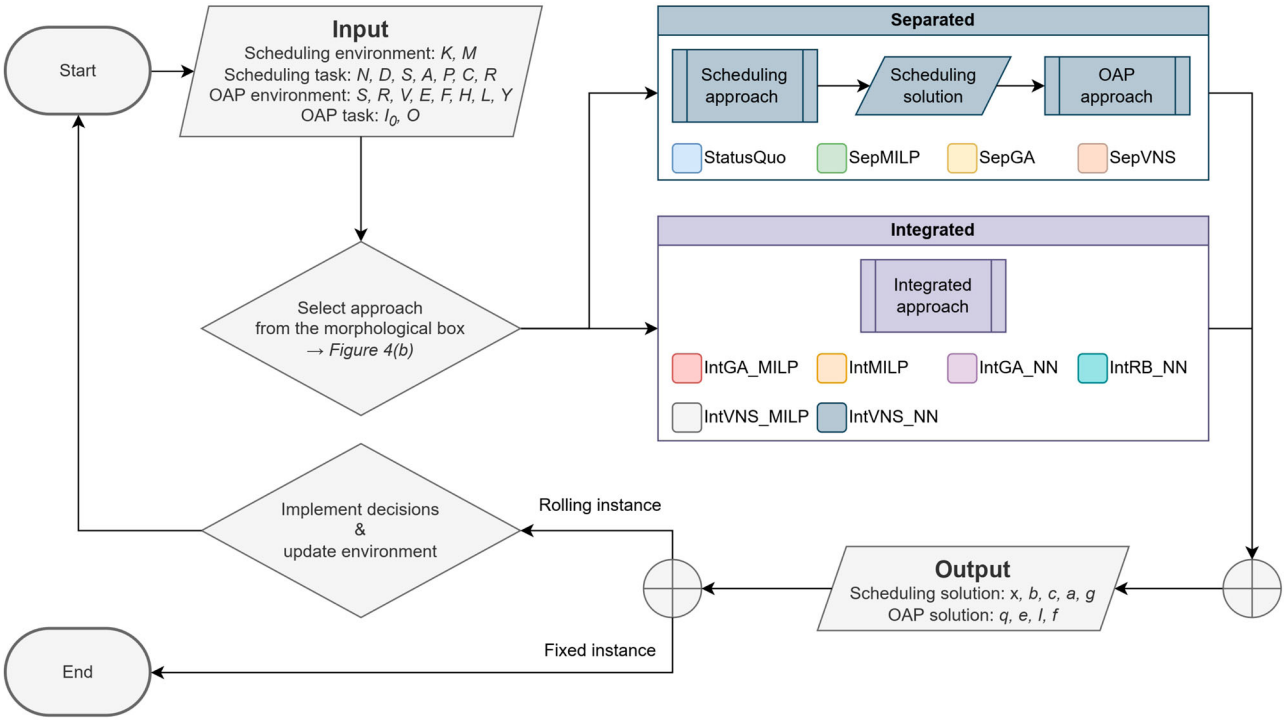
The SepGA approach is the third method that solves the scheduling and procurement problems sequentially. SepGA is employed on larger problem instances where the scheduling MILP is no longer solvable, rendering SepMILP infeasible. SepGA first solves the scheduling problem using the scheduling GA as described in Section 4.1.1 and Figure 1, resulting in an approximate tardiness-minimising schedule. SepGA then computes the optimal procurement plan pertaining to this scheduling solution by solving the OAP MILP (see Figure 4(a)). SepGA is the separated counterpart to IntGAMILP and the approximate counterpart to SepMILP.

4.2.4. Separated variable neighbourhood search (SepVNS)

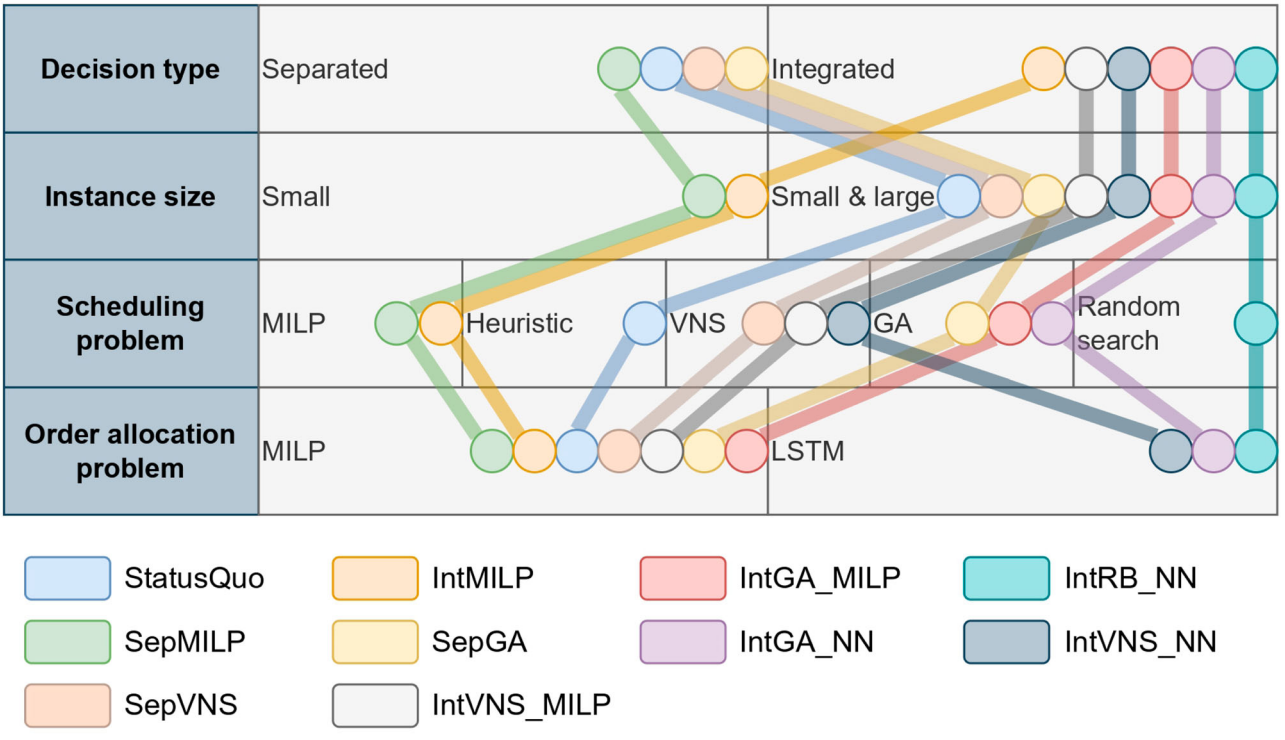
Finally, the SepVNS approach is the fourth and last method that solves the scheduling and procurement problems sequentially. Very similar to SepGA, SepVNS is employed on larger problem instances where the scheduling MILP is no longer solvable, rendering SepMILP infeasible. SepVNS first solves the scheduling problem using the scheduling VNS as described in Appendix 3, resulting in an approximately tardiness-minimising schedule. SepVNS then computes the optimal procurement plan pertaining to this scheduling solution by solving the OAP MILP. Just as SepGA to IntGAMILP, SepVNS is the separated counterpart to IntVNSMILP and the approximate counterpart to SepMILP. Comparing SepVNS to the integrated VNS approaches yields another possibility to compare integrated to separated decision-making and to quantify the benefit of the first.

4.2.5. Integrated mixed-integer linear program (IntMILP)

The IntMILP approach is one of the 6 methods solving the problem in an integrated fashion. IntMILP computes a solution to the integrated problem by solving the integrated MILP (see Section 3.3). Since the scheduling problem itself is NP-hard, so is the integrated problem, hence IntMILP is only feasible on small problem instances. The approach is useful for evaluating our integrated GAs by comparing them to the optimal integrated solution. IntMILP is the integrated counterpart to SepMILP



(a)



(b)

Figure 4. Framework for the separated and integrated solution approaches. (a) The flowchart describes how the integrated and separated solution approaches work. The inputs and outputs of all approaches are identical, i.e. the approaches can be used in a plug-and-play fashion. Approaches framed in dark grey solve the scheduling and procurement problems sequentially, approaches framed in purple construct integrated decisions. On rolling horizon instances, solutions affect future problem instances and (b) The morphological box describes the characteristics of the solution approaches considered in this work. Purely MILP-based approaches provide optimal solutions for their respective (sub)problems, but are only feasible for small instances, while heuristics and GA-based approaches can also solve large instances.

and the exact counterpart to IntGAMILP, IntGANN and IntRBNN.

4.2.6. Integrated genetic algorithm with order allocation mixed-integer linear program (IntGAMILP)

IntGAMILP is one of the five approximate approaches that yield integrated decisions. It is suitable for problem instances of all sizes, including large instances where IntMILP is infeasible. IntGAMILP uses the integrated GA as described in Section 4.1.1, solving the procurement MILP to compute each individual's integrated fitness (see Figures 1 and 4(a)). IntGAMILP is the integrated counterpart to SepGA, the approximate counterpart to IntMILP, and closely related to IntGANN.

4.2.7. Integrated genetic algorithm with LSTM-based neural network (IntGANN)

The IntGANN method is the second approximate approach that solves the problem in an integrated way. IntGANN uses the integrated GA as described in Section 4.1.1 (see also Figure 1). For the fitness computation, IntGANN relies on the procurement cost approximation of the LSTM-based neural network as described in Section 4.1.2. This implies that the individuals do not hold information about the optimal procurement plan, but only about the approximate optimal cost. Therefore, the OAP MILP is solved once for the final best individual to obtain a complete solution, i.e. one that includes both a scheduling and a procurement plan. The approach is particularly well-suited for large instances, as the real-time prediction of procurement costs saves valuable computation time that can be allocated to the GA's search effort. IntGANN is an integrated counterpart to SepGA, the approximate counterpart to IntMILP, and closely related to IntGAMILP differing only in the way the fitness values are computed. To better contextualise its performance, it can be compared to the random baseline IntRBNN and the more sophisticated baseline IntVNSNN.

4.2.8. Integrated random baseline with LSTM-based neural network (IntRBNN)

Without an integrated baseline approach, it would be difficult to assess the performance of our proposed approach, IntGANN. In the absence of a readily applicable integrated baseline from the literature, we introduce an integrated random search baseline, IntRBNN, to contextualise the performance of IntGANN. Within the computational time limit, IntRBNN repeatedly constructs solutions to the scheduling problem at random (i.e. random permutations of the job IDs and a random delay vector), evaluates them by calculating their tardiness cost analytically and approximating their procurement cost using the LSTM-based neural network, and keeps track of the individual with the lowest approximate total cost. Finally, as with IntGANN, the OAP MILP is solved once for the best encountered individual to provide a complete scheduling and procurement plan. IntRBNN is an approximate counterpart to IntMILP and is primarily designed for comparison with IntGANN and IntVNSNN.

4.2.9. Integrated variable neighbourhood search with order allocation mixed-integer linear program (IntVNSMILP)

IntVNSMILP is one of the two approximate integrated approaches employing the VNS of Thevenin, Zufferey, and Glardon (2017). It is suitable for problem instances of all sizes, including large instances where IntMILP is infeasible. IntVNSMILP is closely related to IntGAMILP, only replacing the GA as the integrated search algorithm by a VNS. IntVNSMILP is the integrated counterpart to SepVNS, the approximate counterpart to IntMILP, and closely related to IntVNSNN and IntGAMILP.

4.2.10. Integrated variable neighbourhood search with LSTM-based neural network (IntVNSNN)

Finally, the IntVNSNN method is the last approximate approach that solves the problem in an integrated way. It is very closely related to IntGANN, only replacing the GA as the integrated search algorithm by a VNS. For the fitness computation, IntGANN relies on the procurement cost approximation of the LSTM-based neural network as described in Section 4.1.2. IntVNSNN is an integrated counterpart to SepVNS, the approximate counterpart to IntMILP, and closely related to IntVNSMILP and IntGANN, differing from the first only in the way the fitness values are computed and to the second only by the search algorithm. To better contextualise its performance, it can be compared to the random baseline IntRBNN.

5. Empirical evaluation

The empirical evaluation is based on a real case study of a PCB manufacturer in central Germany facing problems in order allocation and scheduling. In Subsection 5.1, we describe the production environment, the available data and the manufacturer's requirements for a decision support model. We then briefly discuss the implementation and configuration of the model in Subsection 5.2, before comparing our approach to other approaches on different instances in Subsection 5.3.

5.1. Production environment & data

Expert interviews with managers at the PCB manufacturer revealed that separate procurement and production

planning does not produce effective and robust plans. Instead, a new decision support model is required to generate an integrated plan for procurement and production. An important requirement for the decision support model is a reasonable run time that allows the current plan to be updated before each shift. A maximum run time of 30 min is considered reasonable in this case. The continuous updating of the plan on a rolling horizon basis is necessary due to the vulnerable position of the supply chain in which the company operates. The challenging intermediate position between large companies with strong market positions leads to longer lead times from suppliers and potential contractual penalties from customers due to delays. In addition, the model must be able to handle large instances of up to 50 jobs and multiple materials and suppliers to match the company's operational environment.

The production system at the company is a two-stage hybrid flow shop with multiple parallel machines (see Figure 5). The first stage includes four surface mount technology (SMT) placement machines that install surface-mounted devices on PCBs. Given the highly customer-specific and variant-rich nature of PCB assembly, the company follows a make-to-order strategy. It organises variants into product families and allocates one setup trolley to each family to reduce setup time. Each trolley is equipped with all the materials required to produce any variant within its family. The limited number of setup trolleys is represented in constraint (2m). When switching from one product family to another, a major setup time of 65 min is required due to the need to change the setup trolley. Conversely, if consecutive jobs belong to the same product family, only a minor setup time of 20 min is required. This differentiation in setup times based on job sequence is known in the literature as sequence-dependent setup times. The second stage of the production system comprises five automatic optical inspection (AOI) machines that visually inspect the assembled PCBs for defects. The setup time between two jobs at this stage is consistently 25 min, regardless of the product family involved.

Figure 6 shows the distribution of job processing times in minutes in both stages for a sample of 652 real jobs. The distributions for both stages are heavily skewed to the right, with lower processing times occurring much more frequently. The processing times in the two stages are highly correlated, meaning that jobs that require a long processing time in one stage tend to take a long time in the other. In our evaluation, jobs are drawn uniformly at random from the visualised job pool and we assume that there are no jobs in progress at the beginning of the planning horizon.

In order for our results to be generalisable across a wide range of cost structures, we evaluate our approach on a large number of different cost scenarios. Since emergency and tardiness costs have the most impact on the problem (see Section 5.3), we test low, medium and high settings for these two cost sources. To keep the computational effort manageable, we only distinguish between a low and a high setting for the remaining cost components (fixed ordering, variable ordering, and inventory holding). Considering all possible combinations of cost levels results in 72 different cost scenarios. Table 4 gives an overview of the cost ranges. The specific cost factors

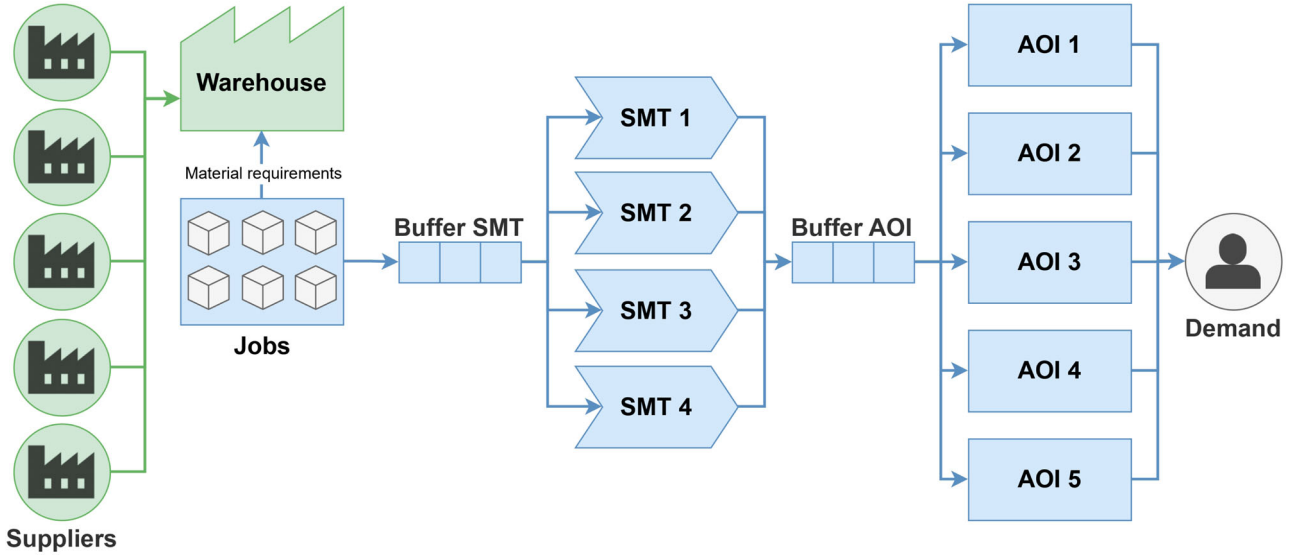


Figure 5. The environment considered in the case study includes a production stage characterised by a hybrid flow shop (depicted in blue) and a procurement stage that involves multiple suppliers and a warehouse (depicted in green). The connection between both stages is the material requirements of the jobs to be scheduled which are taken from the warehouse.

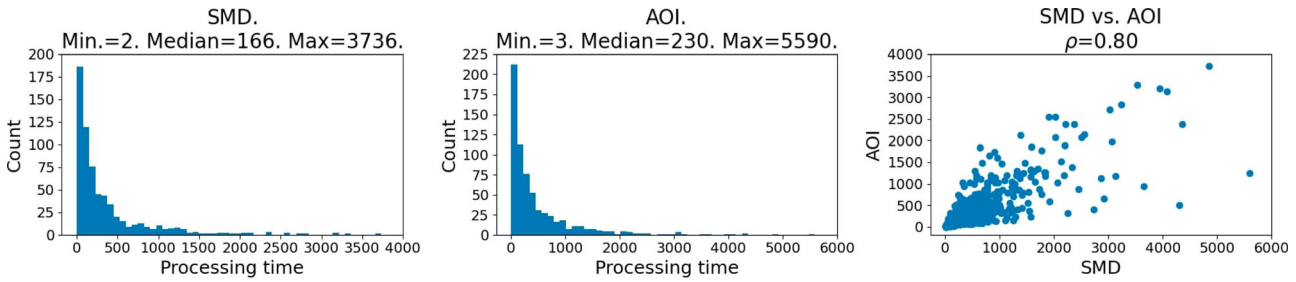


Figure 6. Distribution and stage-wise correlation (ρ) of job processing times in minutes for 652 sample jobs.

Table 4. Overview of cost scenarios. The specific cost factors are drawn uniformly at random from the given ranges.

Cost component	Low	Medium	High
Fixed ordering	300	–	1000
Variable ordering	[10, 15)	–	[20,50)
Emergency ordering	[60,80)	[80,100)	[100,200)
Inventory holding	[1,3)	–	[5,10)
Tardiness	[1/60, 5/60)	[10/60, 20/60)	[30/60, 50/60)

(e.g. emergency cost value $e_{s,r}$ for procuring one unit of raw material r from supplier s) are drawn uniformly at random from these ranges. Furthermore, while the scheduling problem works with minutes, the OAP works with days. For the integrated problem, we use a time conversion factor of $G = 480$, corresponding to a working day of eight hours. The raw material requirements per job range from zero to nine units, i.e. $\alpha_{j,r} \in [0, 9]$. Whether supplier s produces raw material r (see variable Y in Section 3.1) is drawn uniformly at random under the condition that every material must be offered. Both starting inventory I_0 and incoming orders O are set to 0 at the beginning of the evaluation.

We also distinguish between four types of instances. In order to compare our approximate solution methods with the optimal solution, we use small instances on which all three MILP formulations are solvable in acceptable time. To test the performance of our approximate methods on problems of realistic size, we rely on larger instances. This two-step evaluation approach is common practice in the literature (Farghadani-Chaharsooghi et al. 2022; Thevenin, Zufferey, and Glardon 2017). As pointed out in Section 2.3, rolling horizon planning is of fundamental importance in industrial supply chain management, but its application remains scarce in the literature (see Section 2.2 and Table 1). To be in line with the relevant literature but also foster the adoption of rolling horizon evaluation, we analyse our approaches on both fixed and rolling horizon instances. An overview of the four different instance types and their impact on external model parameters is provided in Table 5. In total, our four instance types and 72 cost scenarios amount to a total of 288 instances on which we evaluate our approaches.

Table 5. Instance types. The specific lead time $L_{s,r}$ for supplier s and raw material r is drawn uniformly at random from the given range.

Instance size	Small		Large	
Horizon type	Fixed	Rolling	Fixed	Rolling
Horizon length (days)	1	15	1	15
New jobs / day	10	3	50	8
Suppliers	3	3	10	10
Raw materials	5	5	15	15
Product families	2	2	25	25
Lead time (days)	[1,3)	[1,3)	[2,6)	[2,6)

Finally, we need to generate training data for the LSTM networks. Each training data sample is an input-output pair, where the input consists of the procurement cost parameters and the material requirement matrices, while the target output is the optimal procurement cost (see Figure 3). In practice, training data would consist of past OAP instances faced by the company. As we do not have access to this data, it is generated artificially to replicate this idea by running IntGAMILP and storing the procurement problems encountered. The idea that training data corresponds to historical OAP instances is simulated by using both different cost parameters and different jobs than during evaluation. With this approach, we generate 7,200,000 input-output samples, which we split into training, validation and test sets using a 60/20/20 split. We train one neural network per instance type, i.e. four networks in total, but use the same neural network for all 72 cost scenarios.

5.2. Implementation and configuration

All of our code is implemented in Python 3.9.5. Our neural networks are implemented using `keras` 2.8.0, the MILPs are solved using Gurobi optimiser version 9.5.0 with a free academic licence and accessed via the Python interface `gurobipy`. Our experiments are run on a Debian 12-operated computing server owned by the University of Mannheim with 1024 GB RAM and 96 CPU threads, which were used to parallelise the large number of combinations of approaches (10), instance types

(4) and cost scenarios combinations (72) (each combination is run on one single core). Our analysis is conducted in Python, the results are visualised using the Python package `matplotlib` 3.5.3.

Our GA approaches, our VNS methods and our neural networks are subject to several hyperparameters. The parameters for the latter are tuned manually to keep the computational effort at a minimum, because manual tuning was sufficient to achieve acceptable results, and because the performance proved to be quite robust with respect to most parameters. We use an architecture of [32, 16, 16] hidden nodes with a `tanh` activation function for the LSTM module, and [32, 16] and [16, 16] hidden nodes for the cost-related Dense and the joint Dense module, respectively, each with a `relu` activation. The networks are trained using a learning rate of 0.001 and a batch size of 1024. Training is terminated after a maximum number of epochs of 10 or stopped early if the validation set loss is not improved for 5 consecutive epochs to prevent overfitting. We employ a mean squared error loss and the Adam optimiser. The hyperparameters of our GA and VNS methods are tuned using the Taguchi method (Berrichi et al. 2010; Fraley et al. 2006; Muthana and Ku-Mahamud 2023; Sarrafha et al. 2015). Details of the tuning process are presented in Sections A.1 (for the GAs) and 3 (for the VNS methods) of the appendix. As a result, we run the integrated GAs IntGAMILP and IntGANN with a population size of 74, a crossover probability of 0.8, a mutation probability of 0.7, an elite size of 2 and a tournament size of 4. The ideal settings for SepGA were found out to be 150, 0.9, 0.5, 6, and 4, respectively. To ensure the comparability of our results, the search heuristics (all GAs, VNSs, and IntRBNN) are terminated after a pre-defined runtime, which we set to 5 min for the small instances, and 30 and 10 min for the large fixed and large rolling instances, respectively. These bounds are derived from the requirements of the company considered in this case study.

5.3. Results

In this section, we present the results of our empirical evaluation. The central objective of our problem is the minimisation of total cost. To that aim, we define the percentage cost gap of an approach A towards a reference approach B as:

$$CostGap(A, B) = \frac{TotalCost(A) - TotalCost(B)}{TotalCost(B)} * 100$$

Table 6 shows the percentage cost gap of each approach by instance size and horizon type, averaged over all 72 cost scenarios. (Detailed cost gaps by cost scenario are available in `KISync_V1_Data` at `https://github.com/xbeier/KISync_V1_Data`.) As reference approach, we choose the column-wise best approach, which is IntMILP for the small instances and IntGANN for the large instances. Four preliminary conclusions can be drawn. First, the StatusQuo baseline performs similarly to SepGA, reflecting the fact that the StatusQuo method is essentially a separated approach. Second, on average, all integrated approaches perform distinctly better than their separated counterparts, with the best integrated approach offering average cost savings between 64.91% and 119.11%. In particular, our proposed integrated approach IntGANN represents a considerable improvement over the current, sequential method StatusQuo. Nonetheless, the magnitude of the savings from implementing integrated decisions strongly depends on the quality of the integrated approach as well as the instance type. E.g. while improving from SepGA to IntGANN yields a cost reduction of 94 percentage points on large, fixed instances, savings amount to only 65 percentage points on large, rolling instances. On the same instance type, switching from SepVNS to IntVNSNN even results in a cost reduction of only 22 percentage points. Third, our proposed integrated approaches IntGAMILP and IntGANN perform reasonably well compared to the optimal solution, with an average cost gap between 6.97% and 14.46% on small instances. Although both approaches have room for further improvement, they are designed to make short-term decisions under limited time, especially on realistically large instances, which requires sacrificing solution quality for computational efficiency. This may also be the reason why, contrary to some of the related literature, we do not observe higher cost savings on larger instances, where the degree of freedom is greater. Finally, our proposed approach IntGANN outperforms both the random baseline IntRBNN as well as the VNS-based baselines IntVNSMILP and IntVNSNN on all instances. While the performance difference to IntRBNN is less

Table 6. Aggregate results. The table shows the percentage cost gaps towards the best approach (in bold) by instance type, averaged over all 72 cost scenarios.

Instance size	Small		Large	
Horizon type	Fixed	Rolling	Fixed	Rolling
SepMILP	90.34	119.11	–	–
SepGA	89.74	82.27	94.23	65.24
SepVNS	89.74	95.36	93.38	58.37
StatusQuo	87.83	82.08	79.33	64.91
IntMILP	**0**	**0**	–	–
IntGAMILP	6.97	12.95	27.03	32.81
IntVNSMILP	32.64	61.75	7.34	20.44
IntGANN	8.86	14.46	**0**	**0**
IntVNSNN	34.59	63.79	9.16	42.93
IntRBNN	23.34	19.44	51.37	51.03

pronounced on small instances, where the search space is small and hence random search performs reasonably well simply because it covers a considerable area, the cost gap increases on large instances, where a smarter search strategy is required. This may also be the reason why the integrated VNS-based methods perform best on the large, fixed variant, which contains the problems with the largest number of jobs.

The fundamental reason why integrated approaches perform better is the conflicting nature of the two objectives, minimising tardiness vs. minimising procurement costs. Figure 7(a) supports this claim by showing a detailed breakdown of total costs into individual cost types. Separated approaches focus on completing jobs on time to reduce tardiness costs, leaving the procurement department to handle material requirements, often resulting in costly emergency orders. In contrast, integrated approaches anticipate this issue and strategically delay jobs, accepting higher tardiness costs in exchange for a more efficient procurement plan. This plan involves larger fixed order, variable order costs and inventory holding costs, but significantly reduces emergency orders. While the integrated VNS-based methods also follow this rationale and the difference in their cost structure compared to the separated approaches is observable, they do not manage to reduce emergency costs quite as much as the other integrated methods. While this saves regular procurement cost such as fixed, variable and holding, this trade-off is suboptimal and the underlying reason for the inferiority of these approaches towards IntGANN. The trade-off between the two objectives is further analysed in Section A.2 of the appendix. The above observations demonstrate that while sequential optimisation is easier to implement, it does not meet the overall objective as effectively as integrated optimisation. As noted by Thevenin, Zufferey, and Glardon (2017), even manual adjustments to sequential solutions cannot match the performance of direct integrated approaches. Figure 7(a) can also explain why StatusQuo slightly outperforms SepGA, and why the latter performs better than SepMILP. StatusQuo and SepGA do not find the optimal, tardiness-minimising scheduling solution, but unknowingly construct a schedule that might be suboptimal with respect to tardiness, but more than offsets the increased tardiness cost in terms of reduced procurement costs.

These observations suggest that integrated decisions are particularly beneficial in cost scenarios with low tardiness and high emergency order costs. Figure 7(b) confirms this claim by visualising the benefit of integrated decisions for different levels of individual cost components. Integrated decisions become more important as emergency costs increase. Conversely, separated decisions become less detrimental as fixed ordering, holding, variable ordering, and tardiness costs increase. The slope of the lines suggests that emergency and tardiness costs have the largest impact on total cost differences, while holding costs have the least.

Table 6 indicates that IntGANN performs particularly well on large instances. Indeed, the gap between IntGANN and IntRBNN increases from small to large instances. Also, IntGANN performs worse than IntGAMILP on small instances but outperforms it on large ones. Similarly, the integrated VNS-based methods improve on larger instances, but for them this effect is much less pronounced. We emphasise that, for comparability, all approaches are terminated after the same, instance-dependent runtime. Under these circumstances, there are two key factors influencing the performance of the integrated heuristics: the number of generations completed (or, in the case of IntRBNN, the number of individuals evaluated), depicted in Figure 8(a), and the accuracy with which the OAP MILP is solved, shown for the neural network in Figure 8(b). While

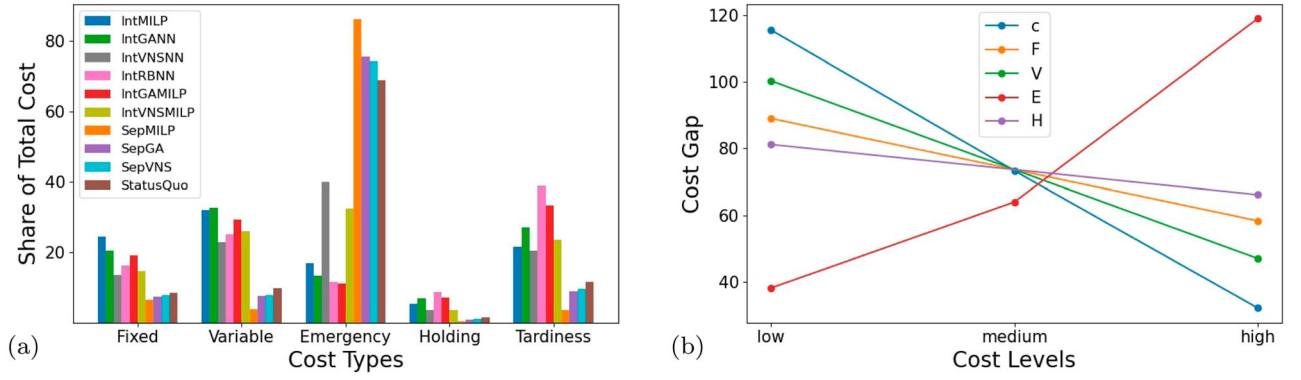


Figure 7. The effect of individual cost components on total cost. (a) Share of cost type in percentage of total costs by approach. The bars show averages over all cost scenarios and instance types and (b) Percentage cost gap of SepGA towards IntGANN by cost type, averaged over all instance types.

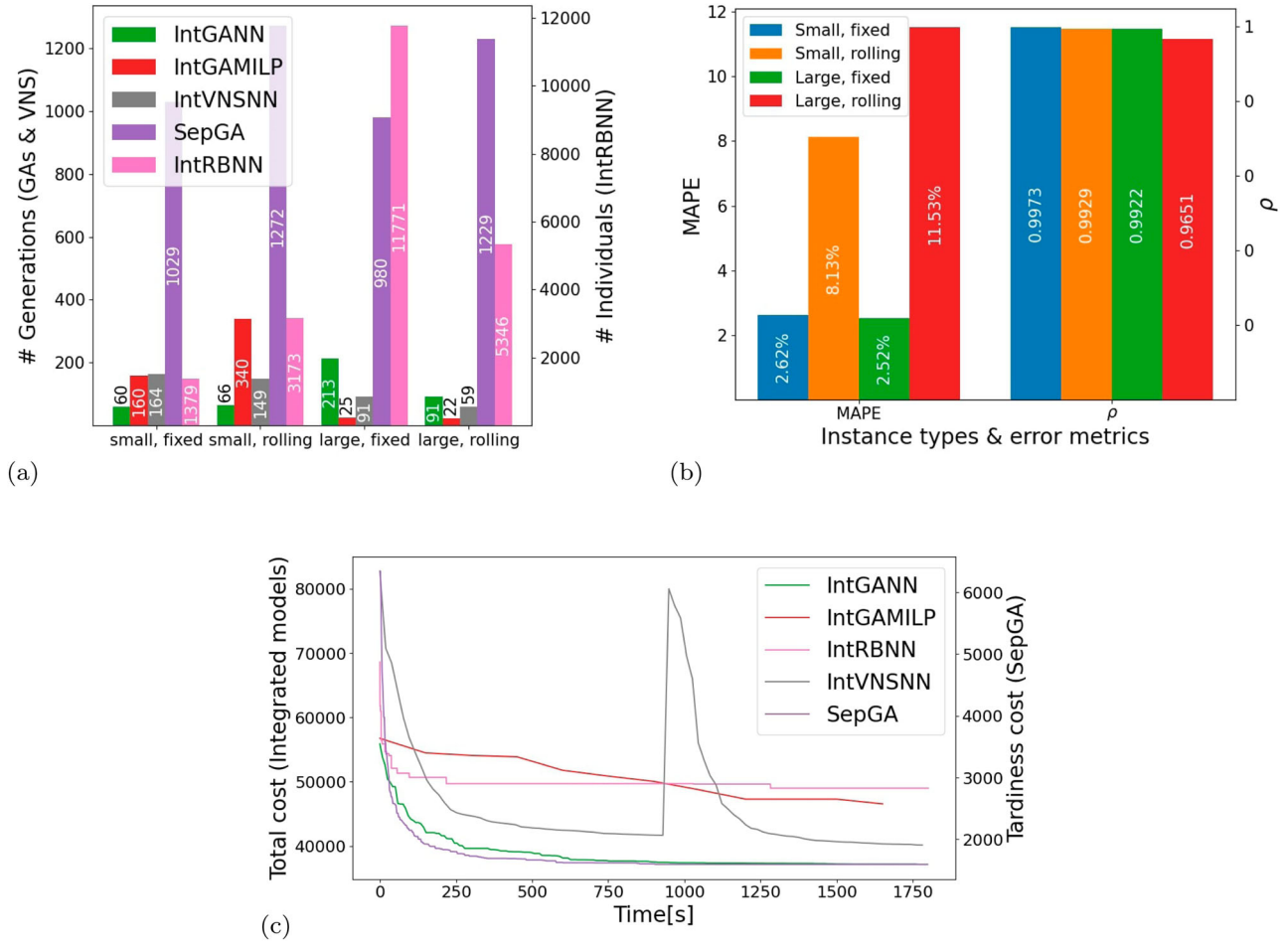


Figure 8. Computational efficiency of the search heuristics. (a) Number of completed generations / evaluated individuals by approach and instance type, averaged over the cost scenarios. (b) Accuracy of the LSTM network in terms of mean average percentage error (MAPE; left y-axis) and correlation with true procurement cost (ρ; right y-axis). (c) Fitness evolution over time for the four search heuristics and one sample cost scenario of the large, fixed type. The integrated approaches minimise total cost (left y-axis), the separated GA minimises tardiness (right y-axis).

IntRBNN evaluates more individuals on large instances due to the longer runtime, it covers a smaller fraction of the search space, reducing its performance. In fact, for n jobs, the search space contains $(n!)^2(max\_delay + 1)^n$ individuals. On small, fixed instances, IntRBNN covers 3189 out of $7.8 * 10^{17}$ instances, i.e. $4.1 * 10^{-13}$% of the search space, which reduces to $\frac{17352}{6.6*10^{152}} = 2.6 * 10^{-147}$% on large, fixed instances. Similarly, on small instances, IntGAMILP completes more generations than IntGANN. This is due to the simplicity of the unrealistically small procurement problems, which Gurobi solves almost instantly. Indeed, for small fixed instances, Gurobi takes on average 0.04 s to solve the MILP formulation of the OAP, while the LSTM network takes 0.07 s. Additionally, IntGAMILP computes the optimal solution to the OAP, whereas IntGANN only approximates the procurement cost, with a small prediction error of 2.62% and 8.13% on small, fixed and small, rolling instances, respectively (see Figure 8(b)). This combination explains the slight gap in performance between IntGANN and IntGAMILP on small instances. On large instances the relationship is reversed. On large, fixed instances, we need to terminate the MILP solver early to allow IntGAMILP to complete at least a few generations (see Figure 8(a)). Early termination results in an average reported MILP optimality gap of 3.48%, slightly larger than the LSTM network error of 2.52%. Coupled with the significantly larger number of generations of IntGANN, this explains its superior performance. On large, rolling instances, the OAP MILPs are smaller and can be solved to optimality in acceptable time. While IntGANN has a significant prediction error of 11.53%, the high correlation between predicted and actual procurement costs of 0.9651 ensures that individuals are still ranked in the correct order. Together with the advantage of a significantly larger number of generations, this

allows IntGANN to dominate in terms of total cost. The VNS-based approaches, on the other hand, do not benefit as distinctly from the larger runtimes granted on large instances. Figure 8(a) demonstrates that, as opposed to IntGANN, IntVNSNN does not complete more generations on large instances. The underlying reason is that GAs operate under a fixed population size regardless of problem complexity, keeping computational effort per population fairly stable across instance types. The VNS-based methods, however, need to traverse a fraction of the incumbent solution's neighbourhood in each generation. As problem size increases, the size of the neighbourhood increases, an therefore so does computational effort, restricting the number of completed generations. Second, we observe that with a correlation of $\rho = 0.8908$ on rolling instances, LSTM accuracy suffers when incorporated in the VNS-based methods, explaining the noticeably bad performance on large rolling instances. Since the LSTM works faultlessly under IntRBNN, where it is subjected to totally random solutions (proving that the LSTM is well-trained on average), we suspect that the neighbourhood definitions employed in the VNS-based methods are suboptimal for our problem and lead to premature convergence in non-optimal, isolated spaces, where the LSTM has difficulties distinguishing the solutions. This idea is supported by Table 6, where we observe that the random baseline outperforms the VNS-based methods on small instances. This demonstrates the advantage of GAs as time-tested approaches on scheduling problems, with well-performing crossover and mutation operators being widely known, whereas VNS seems less straightforward to adapt.

Figure 8(a) also shows that SepGA completes a much higher number of generations in the same computational time as the integrated approaches, due to the simplified fitness computation. However, since tardiness-optimising scheduling plans are not necessarily optimal in terms of total cost, and since the GA converges after some time, SepGA does not benefit from this computational efficiency. The latter point is illustrated in Figure 8(c), which plots the fitness evolution over time for five of the search heuristics for one sample cost scenario of the large, fixed type. While IntGAMILP would benefit from more runtime, IntRBNN, IntGANN and SepGA converge much faster and could be terminated earlier. Indeed, in 95% of the large, fixed cost scenarios, IntGANN has achieved a fitness value within 5% of the optimum after only 15 min, highlighting its ability to make high-quality operational decisions on instances of realistic size in limited time. The figure also demonstrates another issue with the VNS architecture. Similar to the other approaches, the tabu search seems to have converged after approximately half the allowed computation time. Applying the shaking operator does make sense here to induce diversity and escape a local optima. However, the shaking operator seems too extreme, catapulting the search to a solution of approximately the same fitness as the initial one, after which it requires almost the entire remainder of the runtime to re-achieve the earlier fitness level. Eventually, time runs out before IntVNSNN manages to drop to the level of IntGANN.

Finally, we can analyse the robustness of our proposed algorithm. For the purpose of hyperparameter tuning, we evaluate 25 different hyperparameter configurations (see Section A.1 of the appendix for details). Figure 9(c) illustrates the performance of these 25 runs based on their cost gap towards the best run, by GA. With a range of 13.63%, IntGANN proves to be least sensitive to hyperparameters, while the performance of SepGA varies the most in response to changes in the hyperparameters (29.66%). Consequently, while hyperparameter tuning can boost the performance of our approach, it is the most stable among the three GAs. Interestingly, the VNS-based methods demonstrate the exact opposite behaviour (Figure 9(d)). Robustness may be important not only with respect to hyperparameters, but also with respect to cost settings. Figures 9(a) and 9(b) show the performance of six heuristics on all 72 cost scenarios of the large, fixed and large, rolling horizons, respectively. IntGANN outperforms the other approaches on 62 out of 72 scenarios of the large, fixed type, the other 10 scenarios are won by IntVNSMILP (7) and IntVNSNN (3). Of the large, rolling type, there are twelve scenarios where IntGAMILP performs best, two where IntRBNN performs best and one where SepGA performs best. Nonetheless, the performance of IntGANN remains relatively stable with a maximum gap to the optimal approach of 18.61%.

6. Managerial insights

The analysis of integrated procurement and production scheduling highlights several key insights for managerial decision-making in SCM. Integrated decision-making demonstrates significant cost savings over separated approaches. Our findings reveal that integrated methods consistently outperform their separated counterparts. This is evident even with approximate methods, which maintain a reasonably small cost gap to optimality in small instances. However, the performance of integrated approaches is highly dependent on the cost structure. In scenarios with high emergency costs, integrated methods provide substantial benefits by reducing expensive last-minute orders. Conversely, in environments with high tardiness but low emergency costs, the advantages of integration diminish. Managers should analyse their specific

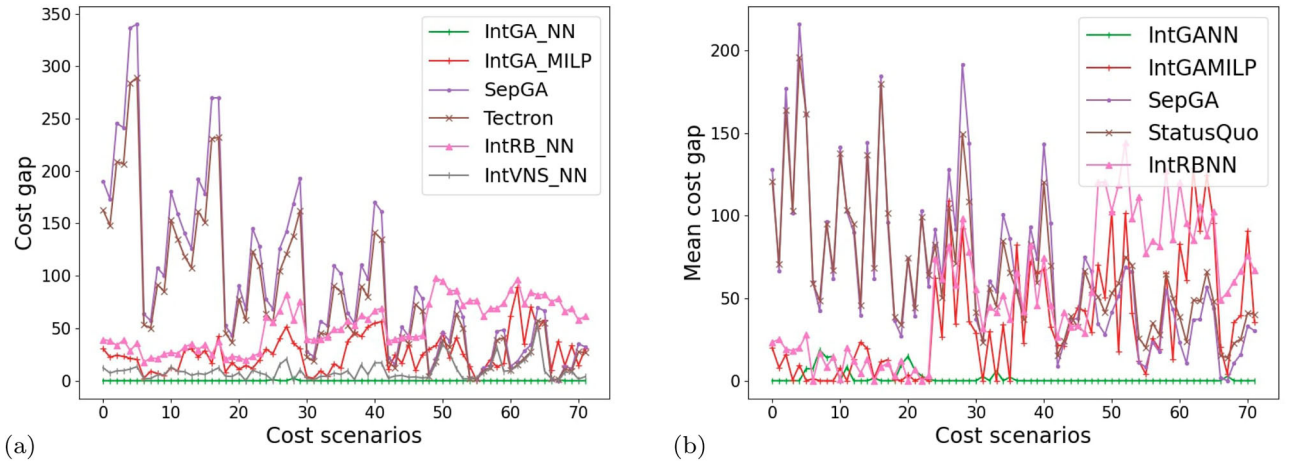


Figure 9. Robustness of the search heuristics with respect to hyperparameters and cost scenarios. (a) Cost gaps for all 72 cost scenarios of the large, fixed type. (b) Cost gaps for all 72 cost scenarios of the large, rolling type. (c) Cost gaps towards optimal run for 25 different hyperparameter configurations by GA. (d) Cost gaps towards optimal run for 25 different hyperparameter configurations by VNS.

cost environment to determine whether the increased complexity of the integrated problem may outweigh the benefits. Thus, it is worthwhile to analyse the cost structure to determine whether the added complexity of integrated methods is justified. The impact of increased complexity is particularly evident when dealing with large and complex instances with many jobs, suppliers and raw materials. In these environments, efficient models and algorithm are required to mitigate the added complexity as much as possible. One potential approach is to use ML-based regression metamodels to learn the input-output-relationship of a more complex model. Such a metamodel can enhance performance despite a higher prediction error.

In conclusion, integrated procurement and production planning offer substantial benefits in terms of cost savings and efficiency. However, managers must consider their specific cost environment, computational resources, and the robustness of prediction models to fully leverage these advantages. To decide whether integrated planning is beneficial, a manager should consider the following guidelines:

(1) Do the isolated objectives have potential for conflict? In our case study, minimising tardiness puts pressure on the procurement department and increases supply costs by requiring expensive emergency orders, which is a substantial reason for the superiority of integrated approaches (see also Section A.2 of the appendix). If isolated objectives are more aligned, integrated decision making is likely to be less beneficial.

(2) Does the cost structure favour integrated decision-making? The higher the emergency ordering costs and the lower the tardiness costs, the greater the potential savings from integrated decisions.

(3) How complex are the procurement and scheduling problems to be solved? The more complex the problems (e.g. many jobs, many suppliers, many raw materials, or long planning horizon), the more challenging it is to find an efficient integrated approach. The magnitude of the benefits from integrated decision-making strongly depends on the quality of the integrated model.

(4) How critical is precision in decision-making within the environment? The more flexible the environment, the easier it is to exploit trade-offs (e.g. to boost the algorithm using ML for the price of minor prediction errors or to allow plans with minor tardiness to reduce the overall costs), making integrated decision-making more beneficial.

7. Conclusions & future work

This study investigated an integrated procurement-production problem, using a real case study to compare the benefits of integrated operational planning versus separated planning. Two research questions guided this analysis: whether integrated planning is beneficial for procurement and production, and how to make decisions efficiently.

The results indicate that integrated decisions in procurement and production lead to significant cost savings, ranging from 65% to 119% compared to separated

planning. This conclusion is drawn from analysing 72 cost scenarios, including rolling horizon scenarios which are still underrepresented in the literature. However, the extent of these savings is highly dependent on the quality of the integrated model, the instance type as well as the cost structure of the problem. Specifically, tardiness costs and emergency costs have the most substantial influence on savings. Integrated planning is particularly beneficial in scenarios with low tardiness costs and high emergency order costs, as jobs can be strategically delayed to optimise procurement plans. Conversely, in cases where the opposite cost conditions prevail, separated planning can sometimes outperform integrated approaches due to the latter's difficulty in finding optimal solutions under tight time constraints. Instance size does not significantly affect the cost savings from integration. On average, integrated decisions lead to 69% cost savings across all scenarios and instance types.

Another aim was to develop an efficient solution approach suitable for short-term decision-making. We proposed a GA combined with a multi-input LSTM-based neural network to generate solutions for instances of realistic size within 30 min. Remarkably, the algorithm reaches a plateau close to the final solution quality within just fifteen minutes (see Figure 8(c)). With mean percentage cost gaps of 9% and 14% the approach performs reasonably close to the optimal solution on small instances. For large instances, the neural network significantly enhances performance, improving results on average by 27% to 33% compared to the exact computation of the OAP and beating both random as well as more sophisticated baselines. By accepting a small prediction error, the neural network significantly improves the computational speed for large instances, allowing considerably more generations to be completed. Given the critical importance of computational efficiency in operational planning, we advocate for the integration of supervised machine learning to accelerate traditional metaheuristics.

Despite these contributions, the study has certain limitations. It does not take into account all the uncertainties that may occur in real supply chains. In particular, at the procurement stage, stochastic lead times and disruptions in the supply network can have a significant impact on the operations of a company. We assume that costs will increase in general when adding stochastic parameters because of higher buffer inventory and slower processing. Future work should aim to create a more realistic environment by including additional external disruptions and other uncertainties such as fuzzy cost and stochastic lead time parameters. In addition, the role of inventory as a buffer between stages needs to be investigated under stochastic conditions. This will require more robust decision-making models, as event-based re-planning alone is not sufficient. Given the success of combining a search heuristic with machine learning, we plan to extend this approach by incorporating an intelligent reinforcement learning agent to anticipate and mitigate stochastic disruptions. In our current study, our metaheuristic of choice was a GA, which is justified by its popularity in the related literature and our own results. Nonetheless, there is a vast body of metaheuristics to choose from and an ongoing debate about their advantages and disadvantages. In the future, it will be interesting to see how other heuristics perform and compare to our proposed approach. We also acknowledge that our current work addresses only procurement and production, two of the three central operational areas of a company. Further research is needed to develop a holistic approach that also integrates operational distribution decisions. Finally, in complex OR decision problems, companies often face multiple competing objectives that cannot always be distilled into a single objective function. Future research should explore the value of integrated operational decisions for multi-objective problems.

Acknowledgments

The authors confirm contribution to the article as follows: **Alexander Bubak**: conceptualisation (models, solution approaches); methodology; software (scheduling problem, MILP, LSTM, evaluation, analysis); writing – original draft (solution modules & solution approaches, empirical evaluation, managerial insights, conclusion, appendix). **Benjamin Rolf**: conceptualisation (research idea, models, solution approaches); methodology; software (baseline, OAP, GA), writing – original draft (introduction, literature review, problem description, empirical evaluation, managerial insights, conclusion), funding acquisition. **Tobias Reggelin**: conceptualisation (research idea); project administration, funding acquisition. **Sebastian Lang**: conceptualisation (scheduling problem). **Heiner Stuckenschmidt**: conceptualisation (research idea), project administration, funding acquisition.

Data availability statement

The data that support the findings of this study are openly available in `KISync_V1_Data` at `https://github.com/xbeier/KISync_V1_Data`.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

This work was supported by the Federal Ministry of Education and Research of Germany (BMBF) under Grant 16DKWN092A.

Notes on contributors

Alexander Beier is a researcher and PhD candidate at the Chair of Artificial Intelligence of the University of Mannheim. He holds a master's degree in Mathematics for Business and Economics from the University of Mannheim. His research interests include artificial intelligence and supply chain optimisation. In his PhD, he currently focuses on the application of machine learning to optimise inventory systems.

Benjamin Rolf is a researcher and PhD candidate at the Institute of Logistics and Material Handling Systems at Otto von Guericke University Magdeburg. He holds a master's degree in Industrial Engineering with a specialization in Logistics from the same university. His primary research interests include the application of machine learning, simulation, and network science in supply chain management. His PhD thesis focuses on the dynamic reconfiguration of supply chains in response to disruptions.

Tobias Reggelin is a research scientist at Otto von Guericke University Magdeburg, where he leads a working group focussed on modeling and simulation in production and logistics. His work includes the development and application of production and logistics management games. Tobias Reggelin earned his doctoral degree in engineering from Otto von Guericke University Magdeburg. He also holds a master's degree in Engineering Management from Rose-Hulman Institute of Technology in Terre Haute, IN, and a diploma in Industrial Engineering and Logistics from Otto von Guericke University Magdeburg.

Sebastian Lang is an Assistant Professor specialising in AI Applications in Production and Logistics at Otto von Guericke University Magdeburg. Additionally, he serves as a Senior Researcher at the Fraunhofer Institute for Factory Operation and Automation IFF in the Robotic Systems department. His PhD thesis on reinforcement learning methods for production scheduling earned him the Dissertation Prize of Otto von Guericke University Magdeburg and the Promotional Prize of the Association of German Engineers (VDI). His research focuses on the development, application, and analysis of AI, optimisation, modelling, and simulation methods in production and logistics. Sebastian Lang has made significant contributions to the field, with over 40 peer-reviewed, Scopus-listed publications.

Heiner Stuckenschmidt is full Professor for Artificial Intelligence at the University of Mannheim. He received his PhD in Computer Science from the Vrije Universiteit Amsterdam and is author of more than 250 peer-reviewed publications. His group is conducting fundamental and applied research in AI including the application of Machine Learning in Inventory Management, Process Analysis and Procurement.

ORCID

Alexander Bubak http://orcid.org/0000-0003-0362-5500
Benjamin Rolf http://orcid.org/0000-0002-5454-8894
Tobias Reggelin http://orcid.org/0000-0003-3001-9821
Sebastian Lang http://orcid.org/0000-0003-3397-1551
Heiner Stuckenschmidt http://orcid.org/0000-0002-0209-3859

References

Alavidoost, M. H., A. Jafarnejad, and Hossein Babazadeh. 2021. "A Novel Fuzzy Mathematical Model for An Integrated Supply Chain Planning Using Multi-Objective Evolutionary Algorithm." *Soft Computing*25 (3): 1777–1801. https://doi.org/10.1007/s00500-020-05251-6.

Benbouja, Mouad, Achraf Touil, Abdelwahed Echchatbi, and Abdelkabir Charkaoui. 2021. "Supply Chain Integration within Mass Customization: Tactical Procurement, Production and Distribution Modeling." *Journal of Industrial Engineering and Management* 14 (2): 250–268. https://doi.org/10.3926/jiem.3182.

Bengio, Y., P. Simard, and P. Frasconi. 1994. "Learning Long-Term Dependencies with Gradient Descent is Difficult." *IEEE Transactions on Neural Networks* 5 (2): 157–166. https://doi.org/10.1109/72.279181.

Berrichi, A., F. Yalaoui, L. Amodeo, and M. Mezghiche. 2010. "Bi-Objective Ant Colony Optimization Approach to Optimize Production and Maintenance Scheduling." *Computers & Operations Research*37 (9): 1584–1596. https://doi.org/10.1016/j.cor.2009.11.017.

Bilgen, B., and H.-O. Günther. 2010. "Integrated Production and Distribution Planning in the Fast Moving Consumer Goods Industry: A Block Planning Application." *OR Spectrum* 32 (4): 927–955. https://doi.org/10.1007/s00291-009-0177-4.

Cao, Wei, and Xifu Wang. 2022. "A Multiobjective Multiperiod Mixed-Integer Programming Optimization Model for Integrated Scheduling of Supply Chain under Demand Uncertainty." *IEEE Access* 10:63958–63970. https://doi.org/10.1109/access.2022.3183281.

Chollet, François. 2018. *Deep Learning with Python*. Shelter Island, NY: Manning Publications.

Darvish, Maryam, and Leandro C. Coelho. 2018. "Sequential versus Integrated Optimization: Production, Location, Inventory Control, and Distribution." *European Journal of Operational Research* 268 (1): 203–214. https://doi.org/10.1016/j.ejor.2018.01.028.

Davis, Lawrence. 1991. *Handbook of Genetic Algorithms*. New York: Van Nostrand Reinhold.

Eiben, A. E., and J. E. Smith. 2015. *Introduction to Evolutionary Computing*. 2nd ed. Natural Computing Series, Berlin, Heidelberg: Springer. Hardcover ISBN: 978-3-662-44873-1

(Published: 10 July 2015); Softcover ISBN: 978-3-662-49985-6 (Published: 17 October 2016); eBook ISBN: 978-3-662-44874-8 (Published: 01 July 2015). https://doi.org/10.1007/978-3-662-44874-8.

Fahimnia, Behnam, Reza Zanjirani Farahani, Romeo Marian, and Lee Luong. 2013. "A Review and Critique on Integrated Production–distribution Planning Models and Techniques." *Journal of Manufacturing Systems* 32 (1): 1–19. https://doi.org/10.1016/j.jmsy.2012.07.005.

Farghadani-Chaharsooghi, Pedram, Pooria Kamranfar, Moha mmad Seyed Mirzapour Al-e Hashem, and Yacine Rekik. 2022. "A Joint Production-Workforce-Delivery Stochastic Planning Problem for Perishable Items." *International Journal of Production Research* 60 (20): 6148–6172. https://doi.org/10.1080/00207543.2021.1985736.

Fleischmann, Bernhard, Herbert Meyr, and Michael Wagner. 2008. *Supply Chain Management and Advanced Planning, Concepts, Models, Software, and Case Studies*, Chap. Advanced Planning, 81–106. Berlin, Heidelberg: Springer.

Fraley, S., M. Oom, B. Terrien, and J. Z. Date. 2006. *Design of experiments via Taguchi methods: Orthogonal arrays*. USA: The Michigan Chemical Process Dynamic and Controls Open TextBook.

Goldberg, David E. 1989. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Boston, MA: Addison-Wesley.

Griffis, Stanley E., John E. Bell, and David J. Closs. 2012. "Metaheuristics in Logistics and Supply Chain Management." *Journal of Business Logistics* 33 (2): 90–106. https://doi.org/10.1111/j.0000-0000.2012.01042.x.

Gu, Jinwei, Manzhan Gu, and Xingsheng Gu. 2015. "A Mutualism Quantum Genetic Algorithm to Optimize the Flow Shop Scheduling with Pickup and Delivery Considerations." *Mathematical Problems in Engineering* 2015:1–17. https://doi.org/10.1155/2015/387082.

Hochreiter, Sepp, and Jürgen Schmidhuber. 1997. "Long Short-Term Memory." *Neural Computation*9 (8): 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735.

Holland, John H. 1992. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. Cambridge, MA: The MIT Press. https://doi.org/10.7551/mitpress/1090.001.0001.

Hrabec, Dušan, Lars Magnus Hvattum, and Arild Hoff. 2022. "The Value of Integrated Planning for Production, Inventory, and Routing Decisions: A Systematic Review and Meta-Analysis." *International Journal of Production Economics* 248:108468. https://doi.org/10.1016/j.ijpe.2022.108468.

Javorcik, Beata. 2020. "Global Supply Chains will Not Be the Same in the Post-COVID-19 World." In *COVID-19 and Trade Policy: Why Turning Inward Won't Work*, edited by Richard E. Baldwin and Simon J. Evenett, 111–116. London: Centre for Economic Policy Research (CEPR).

Jungwattanakit, Jitti, Manop Reodecha, Paveena Chaovalitwongse, and Frank Werner. 2008. "Algorithms for Flexible Flow Shop Problems with Unrelated Parallel Machines, Setup times, and Dual Criteria." *The International Journal of Advanced Manufacturing Technology* 37 (3-4): 354–370. https://doi.org/10.1007/s00170-007-0977-0.

Krajewski, Lee, and Jerry C. Wei. 2001. "The Value of Production Schedule Integration in Supply Chains." *Decision Sciences* 32 (4): 601–634. https://doi.org/10.1111/j.1540-5915.2001.tb00974.x.

Kumar, Ramesh, L. Ganapathy, Ravindra Gokhale, and Manoj Kumar Tiwari. 2020. "Quantitative Approaches for the Integration of Production and Distribution Planning in the Supply Chain: A Systematic Literature Review." *International Journal of Production Research* 58 (11): 3527–3553. https://doi.org/10.1080/00207543.2020.1762019.

Miller, Brad L., and David E. Goldberg. 1995. "Genetic Algorithms, Tournament Selection, and the Effects of Noise." *Complex Syst.* 9:193–212. https://api.semanticscholar.org/CorpusID:6491320

Moons, Stef, Katrien Ramaekers, An Caris, and Yasemin Arda. 2017. "Integrating Production Scheduling and Vehicle Routing Decisions at the Operational Decision Level: A Review and Discussion." *Computers & Industrial Engineering* 104:224–245. https://doi.org/10.1016/j.cie.2016.12.010.

Muthana, Shatha Abdulhadi, and Ku Ruhana Ku-Mahamud. 2023. "Taguchi-Grey Relational Analysis Method for Parameter Tuning of Multi-Objective Pareto Ant Colony System Algorithm." *Journal of Information and Communication Technology* 22 (2): 149–181. https://doi.org/10.32890/jict2023.22.2.1.

Rajwar, Kanchan, Kusum Deep, and Swagatam Das. 2023. "An Exhaustive Review of the Metaheuristic Algorithms for Search and Optimization: Taxonomy, Applications, and Open Challenges." *Artificial Intelligence Review* 56 (11): 13187–13257. https://doi.org/10.1007/s10462-023-10470-y.

Russell, Stuart, and Peter Norvig. 2022. *Artificial Intelligence: A Modern Approach*. 4th ed. Harlow: Prentice Hall.

Sahin, Funda, Arunachalam Narayanan, and E. Powell Robinson. 2013. "Rolling Horizon Planning in Supply Chains: Review, Implications and Directions for Future Research." *International Journal of Production Research* 51 (18): 5413–5436. https://doi.org/10.1080/00207543.2013.775523.

Sarrafha, Keyvan, Seyed Habib A. Rahmati, Seyed Taghi Akhavan Niaki, and Arash Zaretalab. 2015. "A Bi-Objective Integrated Procurement, Production, and Distribution Problem of A Multi-Echelon Supply Chain Network Design: A New Tuned MOEA." *Computers & Operations Research* 54:35–51. https://doi.org/10.1016/j.cor.2014.08.010.

Sawik, Tadeusz. 2016. "Integrated Supply, Production and Distribution Scheduling under Disruption Risks." *Omega* 62:131–144. https://doi.org/10.1016/j.omega.2015.09.005.

Shakhsi-Niaei, Majid, and Atefeh Sajadian. 2021. "Multi-Project and Procurement Scheduling for Manufacturing-To-Order Environments under Price Inflation." *International Journal of Supply and Operations Management* 8 (4): 381–400.

Shirvani, N., and S. Shadrokh. 2013. "Coordination of a Cyclic Three-Stage Supply Chain for Fast Moving Consumer Goods." *Iranian Journal of Operations Research* 4:175–190.

Thevenin, Simon, Nicolas Zufferey, and Rémy Glardon. 2017. "Model and Metaheuristics for a Scheduling Problem Integrating Procurement, Sale and Distribution Decisions." *Annals of Operations Research* 259 (1-2): 437–460. https://doi.org/10.1007/s10479-017-2498-z.

Utama, Dana Marsetiya, Imam Santoso, Yusuf Hendrawan, and Wike Agustin Prima Dania. 2022. "Integrated Procurement-Production Inventory Model in Supply Chain: A Systematic Review." *Operations Research Perspectives* 9:100221. https://doi.org/10.1016/j.orp.2022.100221.

Woolf, Peter2016. *Chemical Process Dynamics and Controls*. The Michigan Chemical Process Dynamic and Controls

Open TextBook. Accessed July 18, 2024, https://libretexts.org/.

Appendix 1. Genetic algorithm

This section provides details about our GA implementation. Algorithm 1 presents pseudocode, Table A1 explains the relevant variables, subroutines and methods.

The algorithm begins by recording the starting timestamp and randomly populates the initial generation of size n. The chromosome of each individual consists of K job sequence vectors and one delay vector (see Figure 2). Since each permutation of the sequence of job IDs to be scheduled in a stage together with any delay vector containing entries in $\{0, 1, \ldots, max\_delay\}$ is a valid solution, the individuals are initialised to random permutations of the job IDs and delay vectors that contain entries from $\{0, 1, \ldots, max\_delay\}$

Algorithm 1 Genetic algorithm

Require: int k, float p_c, float p_m, int n, int e, float *timeLimit*, boolean *integrated*, model *OAPSolver*
Ensure: best individual ind_{best}

$start \leftarrow currentTimestamp()$
$pop \leftarrow randomInit(n)$
while true do
  for *ind* in *pop* **do**
    $ind.solution \leftarrow ind.evaluate(integrated, OAPSolver)$
  end for
  if $currentTimestamp() - start \geq timeLimit$ **then**
  end if
  $elite \leftarrow getBest(pop, e)$
  $parents \leftarrow \emptyset$
  for $i = 1$ to $n - e$ **do**
    $parents \leftarrow parents \cup tournamentSelection(pop, k)$
  end for
  $pop \leftarrow \emptyset$
  for p_1, p_2 in $randomPairs(parents)$ **do**
    $c_1, c_2 \leftarrow crossover(p_1, p_2, p_c)$
    $c_1, c_2 \leftarrow mutation(c_1, p_m), mutation(c_2, p_m)$
    $pop \leftarrow pop \cup \{c_1, c_2\}$
  end for
  $pop \leftarrow pop \cup elite$
end while
$ind_{best} \leftarrow getBest(pop, 1)$
if *integrated* **and** *OAPSolver* is *LSTM* **then**
  $ind_{best}.solution \leftarrow ind.evaluate(integrated, MILP)$
end if
return ind_{best}

Table A1. Overview of *variables*, *methods* and *subroutines* used in the GA.

Name	Domain/inputs	Description
k	$\mathbb{N}_{\geq 1}$	Tournament size
p_c	[0, 1]	Crossover probability
p_m	[0, 1]	Mutation probability
n	$\mathbb{N}_{\geq 1}$	Population size
e	$\{e \in \mathbb{N} : n - e \bmod 2 = 0\}$	Size of the elite
timeLimit	$\mathbb{R}_{\geq 0}$	Time limit (termination criterion)
integrated	{*True*, *False*}	GA variant (scheduling or integrated)
OAPSolver	{*MILP*, *LSTM*}	Subroutine to solve the OAP problem
currentTimestamp	–	Returns the current timestamp
randomInit	n	Randomly initialises a population of size n
evaluate	*integrated*, *OAPSolver*	Computes an individual's fitness value
getBest	*pop*, e	Returns the e best elements of the population *pop*
tournamentSelection	*pop*, k	Selects one individual from *pop* playing a tournament of size k
randomPairs	*parents*	Returns random pairs of individuals from *parents*
crossover	p_1, p_2, p_c	Performs crossover on parents p_1 and p_2 with probability p_c
mutation	c	Performs mutation with probability p_m on child c

uniformly at random. While a termination criterion is not met (in our implementation, for the sake of comparability of approaches, we use a time limit), we repeat the while loop comprising fitness computation, parent selection, and reproduction (see Figure 1). Individuals are evaluated one-by-one. The fitness computation depends on whether the GA solves the scheduling or the integrated problem, as this requires either the tardiness cost only or the sum of tardiness and procurement costs. In the latter case, we further distinguish between computing the exact procurement cost by solving the OAP *MILP* (which also returns the corresponding optimal procurement plan) or predicting a cost estimate using the *LSTM* (see Figure 1). This (initialisation and evaluation for the first generation or, for later generations, parent selection, reproduction and evaluation) completes the current generation, so we check the termination criterion. In case we continue, we retrieve the best e individuals of our population as the elite and initialise the parent set as an empty set. The parent set, which is of size $n-e$ as the remaining e spots are reserved for the elite, is populated by tournament selection based on the current population and tournament size k. Next, we divide the parent set into random pairs (which necessitates an even number of parents, i.e. $n-e$ must be even), perform crossover and mutation, and use the resulting children to form the next generation. Finally, we join the unaltered elite, ensuring the size of the population remains stable at n and the best observed fitness value does not deteriorate, and continue the loop by evaluating the new population. (For computational efficiency, individuals with known fitness values, including but not limited to those in the elite, are not evaluated again.) If the time limit is exceeded, we exit the while loop and retrieve the best individual. In case the integrated problem is solved with the LSTM fitness approximation, this individual does not hold information about the procurement plan (only the approximate cost), so the plan needs to be computed now by solving the OAP MILP (see Figures 1 and 4(a)). Finally, we return the best individual.

Appendix 2. Taguchi analysis

We rely on the Taguchi method to tune the hyperparameters of our GAs. The Taguchi method, originally developed for quality control of manufactured goods, is an efficient statistical approach to investigate the effect of parameters on the mean and variance of a process (Woolf 2016). The experimental design involves so-called orthogonal arrays that prescribe which parameter combinations to test, which allows to collect the desired data with the minimum number of experiments. This idea makes the approach appealing for hyperparameter tuning of optimisation models and is especially popular for metaheuristics like GAs (Berrichi et al. 2010; Muthana and Ku-Mahamud 2023; Sarrafha et al. 2015).

A.1 Hyperparameter tuning

To optimise our three GAs, we test the parameters `population size` $\in \{50150\}$, `crossover probability` $\in \{0.30.9\}$, `mutation probability` $\in \{0.10.7\}$, `size of theelite` $\in \{210\}$, and `tournament size` $\in \{210\}$ for each GA. We use the L25 orthogonal array (Woolf 2016), allowing us to reduce the number of experimental runs from a total of $5^5 = 3125$ to 25. For each run, i.e. hyperparameter configuration, we conduct 72 trials by running the GAs in each cost scenario of the large, fixed type, resulting in a 25×72 results table for each GA. Finally, we convert the achieved objective values (tardiness for the separated, total cost for the integrated GAs) to column-wise cost gaps to allow for the results to be comparable across trials and GAs and compute row-wise means as the final performance indicator of the respective run.

Table A2. Best hyperparameters for each GA.

Approach	Population size	Crossover probability	Mutation probability	Elite	Tournament size
SepGA	150	0.9	0.5	6	4
IntGAMILP	74	0.8	0.7	2	4
IntGANN	74	0.8	0.7	2	4

The best hyperparameters are those minimising the mean cost gap and are identified as reported in Table A2. Together with the rest of this study's data, the detailed results are available at `https://github.com/xbeier/KISync_V1_Data`. Figure A1 summarises the main results by reporting the mean cost gaps averaged over all runs that feature the corresponding hyperparameter value. We observe that IntGAMILP suffers from large population sizes, as this further reduces the number of generations the algorithm can complete. We can also observe that, especially for SepGA and IntGANN, larger crossover and mutation probabilities tend to perform better than low values. The results for the remaining two hyperparameters are less distinct. Elite settings at the extreme perform better than moderate values, while smaller tournament sizes are more appropriate for IntGANN and SepGA and larger ones for IntMILP. Figure A1(f) summarises the effects of each hyperparameter by plotting its cost gap range for each GA. In conclusion, IntGANN is the least sensitive to all hyperparameters except for the mutation probability. While SepGA is especially sensitive to mutation and crossover probabilities, IntGAMILP is rather robust to these parameters and depends more distinctly on population and tournament size.

A.2 Total and procurement cost vs. tardiness cost

The above Taguchi analysis allows us to visualise the relation between tardiness on the one and procurement and total cost on the other hand. Figures A2(a–c) plot procurement cost vs. tardiness cost as achieved by SepGA during the 25 hyperparameter runs, where procurement cost is averaged over all scenarios with low, medium and high tardiness cost, respectively. Figures A2(d–f) show the corresponding plots with mean total cost on the y-axis. Plots (a)–(c) show a distinct negative correlation between procurement and tardiness cost, demonstrating the conflicting nature of the two individual objective functions. Minimising tardiness puts pressure on the procurement department, increasing supply costs by forcing sub-optimal purchases. Plots (d)–(f) demonstrate that this relation holds for total cost, too. I.e. on average, the increase in procurement cost incurred by a more punctual scheduling solution more than offsets the decrease in tardiness costs, explaining the bad performance of separated approaches in our environment.

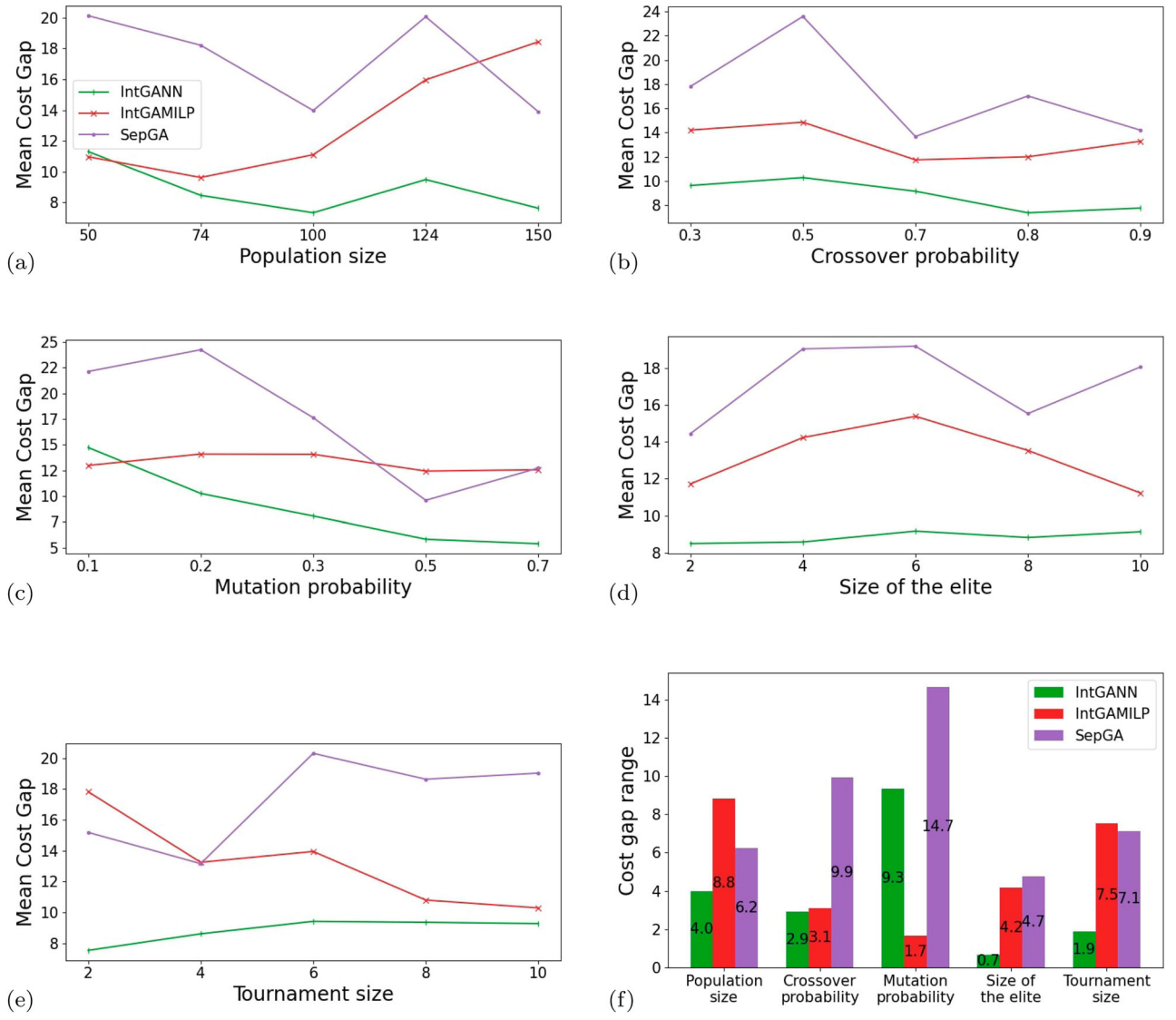


Figure A1. Taguchi analysis for GA-based methods. (a) Effect of the population size on GA performance. (b) Effect of the crossover probability on GA performance. (c) Effect of the mutation probability on GA performance. (d) Effect of the elite size on GA performance. (e) Effect of the tournament size on GA performance. and (f) Hyperparameter importance by GA.

Appendix 3. Baseline algorithm

We use the Reactive VNS algorithm proposed by Thevenin, Zufferey, and Glardon (2017) as a baseline to compare solution quality and computation time. VNS is a metaheuristic optimisation algorithm that works by systematically changing neighbourhood structures within the search space to escape local optima and explore new regions. The algorithm starts with an initial solution and iteratively applies two main phases: shaking, which diversifies the search by randomly selecting a solution from a larger neighbourhood, and local search, which intensifies the search by exploring smaller, more focussed neighbourhoods. In this subsection, we outline the concept of the baseline algorithm and describe how it has been adapted to the problem considered in this article. For a detailed description of the algorithm and its underlying assumptions, we refer to Thevenin, Zufferey, and Glardon (2017).

Thevenin, Zufferey, and Glardon (2017) addressed a single-stage scheduling problem with non-identical parallel machines. The scheduling problem is constrained by precomputed production bounds derived from material availabilities, which are determined by initial inventory levels and regular orders arriving during the planning horizon. For each day, there is a Pareto front of multiple production bounds, including possible emergency orders that may increase the inventory of certain materials but also incur higher costs. Since batch processing is a key concept in their scheduling problem, multiple jobs of the same product type are aggregated into blocks. Consequently, the sequence on each machine cannot contain more than one block per product type. Although splitting similar jobs into multiple blocks is not allowed, each job can be individually rejected and removed from a block if necessary. The algorithm uses two neighbourhood structures, denoted N_{block} and N_{job}, which act as modules for the shaking and local search phases of the VNS. Infeasible solutions that violate the production bounds are repaired by rejecting jobs until feasibility is restored. The following neighbourhoods are defined:

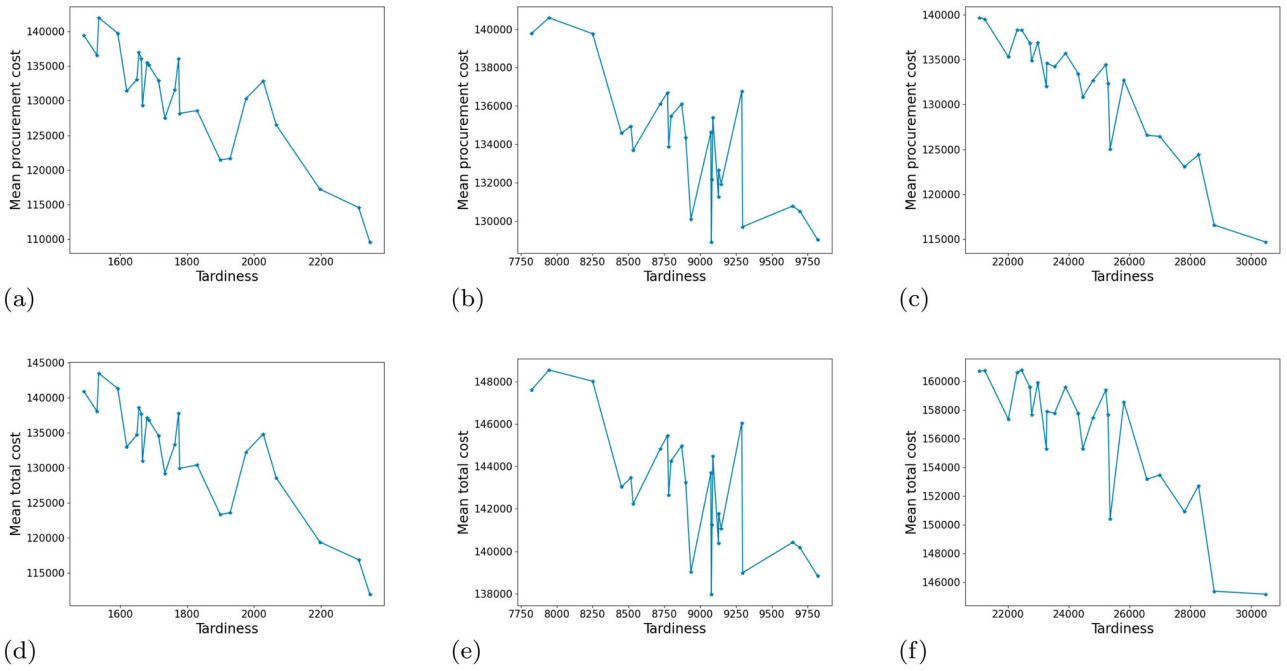


Figure A2. Mean procurement cost and mean total cost vs. tardiness cost for the 25 Taguchi runs of SepGA. (a) Mean procurement cost vs tardiness cost for low tardiness cost scenarios. (b) Mean procurement cost vs. tardiness cost for medium tardiness cost scenarios. (c) Mean procurement cost vs. tardiness cost for high tardiness cost scenarios. (d) Mean total cost vs. tardiness cost for low tardiness cost scenarios. (e) Mean total cost vs. tardiness cost for medium tardiness cost scenarios and (f) Mean total cost vs. tardiness cost for high tardiness cost scenarios.

- N_{block} encompasses all solutions obtained by exchanging the positions of two blocks or shifting a block to a different position within the sequence.
- N_{job} includes all solutions resulting from inserting a previously rejected job into a block or moving a job from one block to another block of the same product type on a different machine. If such a block already exists on the target machine, the job is inserted into it; otherwise, a new block is created at the position that minimises setup costs.

The shaking operator performs n random moves within either N_{block} or N_{job} where $n = i\%$ of the total jobs in the instance. The variable i controls the degree of exploration and is adjusted within the interval $[i_{\min}, i_{\max}]$. This approach leads to increased randomness and exploration as instance sizes and iteration counts grow. The neighbourhood selected for shaking is the one less explored during the local search of the previous iteration. For the local search phase, they employ a Tabu Search algorithm that starts from the shaken solution and terminates after 500 iterations. In order to ensure comparability between our approaches, we need to enforce a stricter time limit. Therefore, we introduce a relative runtime parameter $rel\_time\_ts \in [0, 1]$, such that each tabu search run is terminated after $rel\_time\_ts * max\_time\_vns$, where max_time_vns is the absolute time allowed for the VNS. The Tabu Search explores the union of both neighbourhoods, N_{block} and N_{job}. As the neighbourhoods may be large for instances with many jobs, the parameter n controls the probability that a neighbour will be evaluated ($n = \max(1, \frac{n}{\max_k\{\mathcal{N}^k\}})$). If the solution obtained from the Tabu Search improves upon the current solution, it replaces the current one. Otherwise, the algorithm increases the degree of exploration by updating $i = i * \alpha$ before proceeding to the next iteration.

As our problem environment is fundamentally different from that encountered in Thevenin, Zufferey, and Glardon (2017), we could not adopt their parameter values. Therefore, we used the Taguchi method (see Appendix 2) to tune the parameters. Results and detailed parameter values are provided in Figure A3 and Table A3.

Given the structural similarities between the scheduling problem studied by Thevenin, Zufferey, and Glardon (2017) and the two-stage hybrid flow shop problem considered in this article, we can adapt their original algorithm to our setting. The original neighbourhood structures focus on merging and splitting blocks to achieve optimal trade-offs between minimising setup times and maximising flexibility. The sequence-dependent setup times in our scheduling problem require similar considerations. Consequently, we define a sequence of jobs belonging to the same product family as a block, although we handle these blocks less rigidly than in the original formulation. In contrast to Thevenin, Zufferey, and Glardon (2017), the sequence of jobs within a block is important in our problem because each job is customer-specific and has an individual due date. Therefore, we allow multiple blocks of the same product family per machine. In addition, we adjust the neighbourhood structures so that swapping and inserting blocks is only allowed within the same stage. Apart from these adjustments, N_{block} does not require any further modifications. For N_{job}, we replace the operation of inserting a rejected job into a block with inserting a single job into a block. This is necessary because in our environment the accept-or-reject decision is pre-determined, so that all jobs in the pool must be scheduled. Consistent

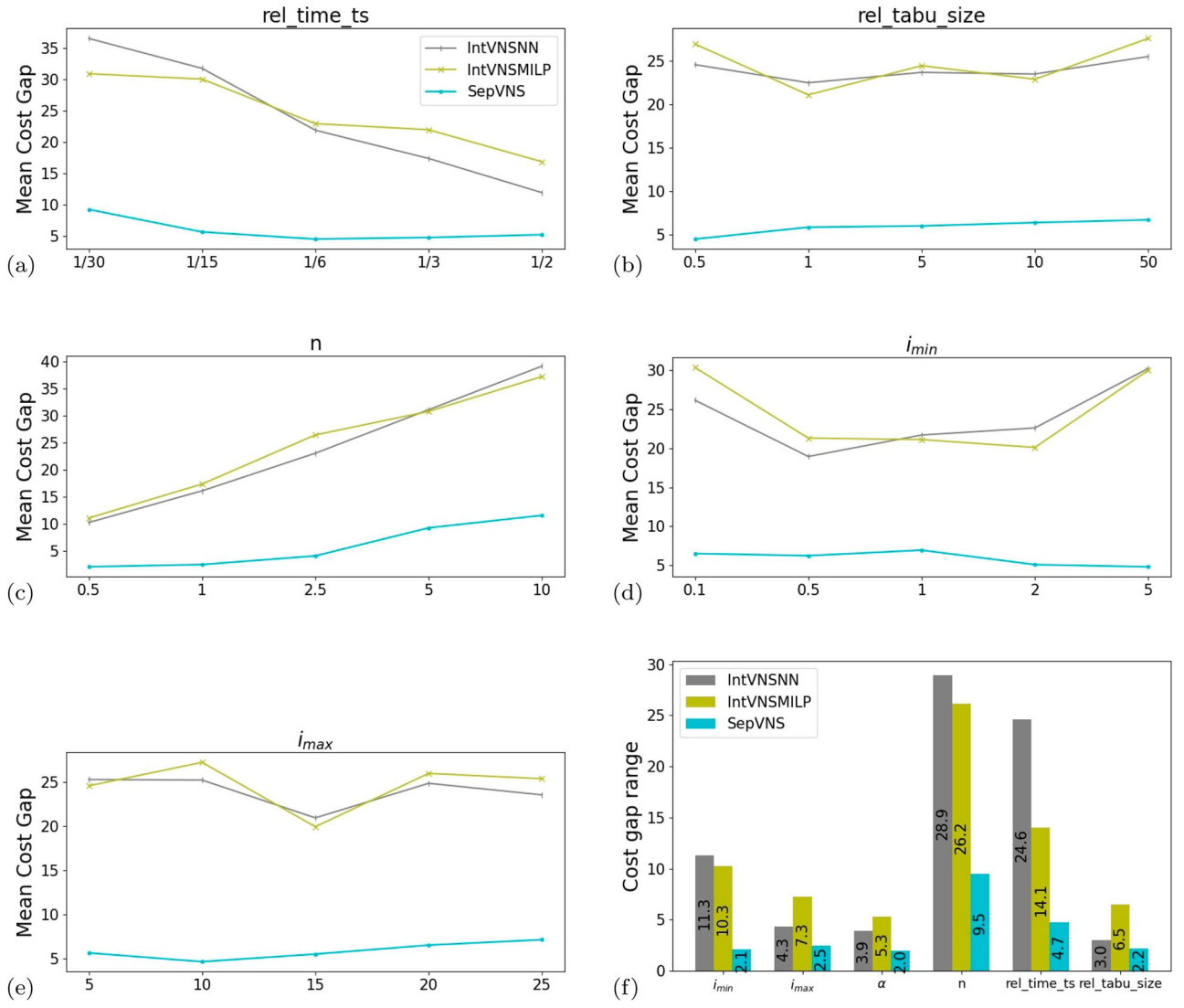


Figure A3. Taguchi analysis for VNS-based methods. (a) Effect of Tabu Search computation time on VNS performance. (b) Effect of tabu list length on VNS performance. (c) Effect of the number of evaluated individuals per tabu iteration on VNS performance. (d) Effect of $i_{\min}$ on VNS performance. (e) Effect of $i_{\max}$ on VNS performance and (f) Hyperparameter importance by VNS method.

Table A3. Configuration of the baseline algorithm for the experiments.

Parameter	Description	SepVNS	IntVNSMILP	IntVNSNN
rel_time_ts	Computation time of Tabu Search	0.5	0.5	0.5
rel_tabu_size	Relative length of tabu list	10	10	5
n	Indicator for number of solutions to evaluate in a Tabu Search iteration	.5	.5	1
$i_{\min}$	Lower bound of exploration variable	5	5	2
$i_{\max}$	Upper bound of exploration variable	15	15	5
α	Balance between exploration and exploitation	1.325	1.325	1.775

with the other algorithms described in Section 4, we use an instance-dependent runtime as the termination criterion.

Thevenin, Zufferey, and Glardon (2017) adopt a fundamentally different approach to integrating procurement and production problems. In their model, inventory levels and material orders are predetermined through tactical planning, with additional materials available only through emergency orders during scheduling. Each material is sourced from a single supplier characterised by deterministic purchase costs and fixed capacities for emergency orders. Moreover, the materials required for each job are unique, which simplifies the identification of the limiting material and allows for the derivation of various production bounds based on the number of emergency orders. In this approach, the regular orders are still decoupled from the material requirements at the production stage because they are pre-determined by the tactical planning. This formulation

significantly reduces the decision space by incorporating the procurement problem merely as additional constraints within the scheduling problem, rather than solving it as a comprehensive optimisation problem. Our procurement environment is more complex with non-unique materials, job-specific material requirements and multiple suppliers per material which disables the calculation of production bounds. This additional complexity disables transferring this approach to our problem because it would require extensive adaptions that would compromise comparability. Therefore, we treat order allocation as a separate optimisation problem that must be synchronised with the scheduling problem. This necessitates determining ordering and scheduling decisions simultaneously, thereby increasing computational complexity. As for the integrated GA-based methods IntGAMILP and IntGANN, we therefore couple the VNS approach with each of the two OAP solution methods, which generates the two integrated VNS-based approaches IntVNSMILP and IntVNSNN.