Optimization of buffer allocations in flow lines with limited supply

Sophie Weiss a, Andrea Matta b and Raik Stolletz a

a Business School, University of Mannheim, Mannheim, Germany; b Department of Mechanical Engineering, Politecnico di Milano, Milan, Italy

ABSTRACT

The supply of flow lines is often assumed to be unlimited or to follow certain distributions. However, this assumption may not always be realistic, as flow lines are usually an integral part of a supply chain where raw material is replenished based on some rule. We therefore include the limited supply into the optimization of buffer capacities in terms of an order policy.

To integrate this type of supply into an optimization model, we exploit the flexibility of a sample-based optimization approach. We develop an efficient rule-based local search algorithm that employs new individual lower bounds in order to determine the optimal buffer capacities of a flow line. In addition to the efficiency of the proposed algorithm, the numerical study demonstrates that the order policy has a significant impact on the optimal buffer allocation.

1. Introduction

1.1. Problem description

Flow lines consist of a number of stations that are arranged in series and separated by buffer spaces. Stochasticity in such lines can be caused by random machine breakdowns, uncertain times to repair, and random processing times. If buffer capacities are limited, blocking and starvation effects may occur. This may lead to a reduction of the throughput of the entire line. Allocating additional buffer capacities decouples the stations and therefore counteracts these effects. However, the average work-in-process in the line increases, which involves additional costs.

The decisions on the total quantity of buffer spaces and their allocation within the flow line, which balance the trade-off between resulting costs and obtained throughput, are known as the Buffer Allocation Problem (BAP). Multiple examples from the practice concerning this problem can be found in the literature. Most examples apply to the automobile industry (Alden et al., 2006; Colledani et al., 2010; Li, 2013), but lines from food industry (Liberopoulos and Tsarouhas, 2002) and other manufacturing applications (Burman et al., 1998; Köse et al., 2015) are also described. These articles demonstrate the potential of operations research methods in determining the optimal allocation of buffer capacities and report on the resulting benefits.

In the literature, the BAP is usually solved under the assumption of unlimited supply (Gershwin and Schor, 2000). To ensure unlimited supply in practice, large inventory levels in front of the first station are required to allow for stochastic effects in the line. Some articles take limited supply into account but assume that the arrival times of the workpieces are exogenously determined (Dallery and Gershwin, 1992; Matta, 2008) or that an additional station models the supply (Dallery and Gershwin, 1992; Helber et al., 2011). However, this is not realistic, as independency of the system state and the arrival pattern is assumed. In reality, orders are placed depending on the inventory level in front of the first station.

1.2. Literature review

Various problem formulations of the BAP with different objectives can be found in the literature. Overviews on these objectives and the existing optimization approaches are given by Gershwin and Schor (2000), Demir et al. (2014), and Weiss et al. (2016). Recent approximations include, among others, metaheuristics, closed-form solutions, gradient algorithms, and quadratic and linear programming. Costa et al. (2015) develop a parallel tabu search that allows simultaneous execution for different seeds. Performance measures are obtained by simulation. Simulated annealing and genetic algorithms are combined by Kose and Kilinci (2015). The genetic algorithm is applied for candidate generation, whereas simulated annealing is used for the acceptance test. Li, Qian, Yang, and Du (2016) apply closed-form solutions and use those to derive lower bounds for the total buffer capacity in the line. Gradient algorithms are presented by Wang et al. (2016) and Shi and Gershwin (2016a). Whereas Wang et al. (2016) use aggregation for performance evaluation, Shi and Gershwin (2016a) apply the DDX algorithm to segments of the line. Chiba (2015) develops a binary search algorithm for a BAP with blocking probabilities as performance measure. In contrast, Li, Qian, Du, and Yang (2016) decompose continuous lines into three-machine systems and use an iterative complete enumeration to approximate the buffer allocation. Alfieri et al. (2016) introduce a column-generation procedure in combination with sampling. A nonlinear sequential quadratic program to jointly calculate the buffer allocation and the
ConWIP level is proposed by Smith (2016). Kolb and Göttlich (2015) combine sequential linear programming with sampling in continuous lines. Liberopoulos et al. (2015) approximate the optimal buffer allocation using shadow prices in the solution procedure of their sample-based linear program.

In the following, we review approaches that provide exact optimal solutions for the BAP or provide bounds on the buffer capacities. Exact analytical results are only available for very small lines under restrictive assumptions (see, e.g., Buzacott (1971)). For longer lines, Matta (2008) proposes a Mixed-Integer Programming (MIP) formulation that uses sampling. Sampling approaches replace the stochastic elements by their sampled counterparts. They therefore allow for a large degree of flexibility. Hence, these approaches can be used for more realistic modeling of the underlying problem. It is possible to allow for any distribution of processing times, times to failure, and repair times, as well as correlations therein. Moreover, the resulting performance measures are sample-exact and converge to the exact value, provided that sample sizes are chosen sufficiently large. However, the corresponding sample-based MIP approach is only capable of solving very small instances with three stations. Alfieri and Matta (2012) introduce the concept of time buffers, which reduce the feasible region of the buffer capacities. Unfortunately, the derivation of the time buffers is only possible for small instances with three stations. Weiss and Stolletz (2015) consider an MIP formulation similar to Matta (2008). To accelerate the solution process, they propose a Benders Decomposition approach in combination with the generation of lower bounds derived from subsystems. They use the flexibility of the approach to demonstrate the impact of correlations on the optimal buffer allocation. The work of Shi and Gershwin (2016b) is closely related, due to the proposed segmentation approach applying the concept of subsystems to estimate the buffer capacities.

Matta et al. (2014) describe a general methodology to derive simulation-optimization models in terms of mathematical programming. This concept is also applied for Base Stock Control Systems and Extended Kanban Control Systems (Pedrielli et al., 2015) and for the optimization of the number of pallets in ConWIP systems (Alfieri et al., 2015).

2. Model of the flow line

This article considers the allocation of a minimum number of total buffer spaces while attaining a pre-defined goal throughput, which is known as the primal BAP (Gershwin and Schor, 2000).

Section 2.1 presents the decision problem and the underlying assumptions for the flow line model. The modeling and the assumptions with respect to the limited supply are explained in detail in Section 2.2.

2.1. Model assumptions and decision problem

The model of the flow line is based on the following assumptions:

- The flow line consists of $m = 1, \ldots, M$ stations in series, which process $W$ workpieces.
- The decision $X_m$ about the capacity of the buffer behind station $m$ is limited by $B_m$.
- An order policy is applied to manage the material supply to the first station; i.e., the supply is limited. Unlimited supply can be modeled by selecting adequate parameters for the policy.
- The buffer behind the last station is infinitely large, $B_M = \infty$. Thus, this station cannot be blocked.
- The processing times of the workpieces at each station are generally distributed.
- The stations may be subject to operation-dependent failures. Time-to-failure and time-to-repair values are generally distributed.
- In the event of blocking, the station finishes the currently processed workpiece. Then, the workpiece waits at the station until a buffer space or the following station becomes available (blocking after service).
- Transportation times through the buffer are insignificant or are already included in the processing times.
- The performance of the line is measured with respect to the expected throughput $E[TH(X_1, \ldots, X_{M-1})]$ and is evaluated under steady-state conditions.

Figure 1 shows an example of a flow line based on these assumptions. The mathematical formulation of the decision problem is given in Equation (1):

$$
\min \sum_{m=1}^{M-1} X_m \\
\text{s.t.} \\
E[TH(X_1, \ldots, X_{M-1})] \geq TH^* \\
X_m \leq B_m, \quad \forall m \\
X_m \geq 0, \quad \text{integer}, \quad \forall m.
$$

The objective function (1a) minimizes the total buffer capacity in the line. Equation (1b) ensures that the goal throughput,
IISE TRANSACTIONS

Figure 1. Flow line under consideration.

\( TH^* \), is attained. In Constraint (1c), the physical floor limitations are defined as upper bounds on the buffer capacities. Equation (1d) ensures that the buffer capacity variables are non-negative and integer.

2.2. Supply of the first station

We assume that the supply of the first station is organized by an order policy that launches replenishment orders depending on the inventory position in front of the first station. The inventory position consists of the current inventory level and already placed orders, which have not yet been received (i.e., these orders did not arrive at the first station). Each order has a predefined lead time of \( T \) time units. Such policies are described by Silver et al. (1998).

We test two types of inventory policies, the \((s, q)\)-order policy and the \((r, S)\)-order policy. However, any type of order policy can be used in our approach.

The \((s, q)\)-order policy is based on a reorder point \( s \), a constant order quantity \( q \), and a lead time \( T \). Whenever the inventory position of the storage in front of the first station drops to the reorder point \( s \) or below, an order of size \( q \) is placed. Consequently, the inventory position must be continuously monitored. Each order requires a lead time of \( T \) periods until delivery.

The \((r, S)\)-order policy, in contrast, is based on periodic review and uses a review interval \( r \), a lead time \( T \), and an order-up-to-level \( S \). An order is placed every \( r \) periods. Each order requires a lead time of \( T \) periods until delivery. The order quantity is chosen in such a way that the order raises the inventory position to the order-up-to-level \( S \). In some cases, it is convenient to always order a multiple of a certain order quantity \( q \) (e.g., truck loads) instead of ordering an arbitrary number of items. In this case, the order quantity is calculated by

\[ \left\lceil \frac{S - \text{inventory position}}{q} \right\rceil \times q. \]

This formula will be used in what follows.

Whenever the inventory in front of the first station is empty, the starting time of the next workpiece is delayed at least until the next order arrives. Otherwise, processing at the first station begins when the previous workpiece leaves the station. Depending on the parameters of the \((s, q)\) or the \((r, S)\) policy, there is a positive probability that the first station will starve, which is not possible when assuming unlimited supply.

This model is closely related to the models proposed in the inventory literature. Axåsäter and Rosling (1993) show that a flow line can be modeled as a series of installations that are supplied by order policies and model the interaction of two consecutive stations. The consideration of the limited supply of the line corresponds to an additional installation in front of the line. However, for these models the waiting and blocking times of the workpieces must be known to determine the lead time of the order policies. This is not the case under general assumptions, as the waiting and blocking times depend on the buffer capacities, and this relation cannot be expressed in a closed form. Therefore, modeling the flow line as a series of installations that are connected by order policies is not applicable. Consequently, approaches from the inventory literature cannot be applied.

3. Individual lower bounds on the buffer capacities

The BAP is an NP-hard problem (Smith and Cruz, 2005). The feasible region grows nonlinearly with the number of stations in the line. This complexity requires solution approaches that extensively reduce the size of the feasible region. Therefore, we develop new lower bounds on the optimal individual buffer capacities in order to reduce the solution space of the BAP.

Weiss and Stolletz (2015) develop Aggregate Lower Bounds (ALBs) for groups of buffers based on the optimization of subsystems of the line. However, using these aggregate bounds, it is not clear how many buffer spaces have to be assigned to which individual buffer.

Individual Lower Bounds (ILBs) restrict the solution space of the BAP more extensively than ALBs. The idea of generating ILBs consists of three steps and is outlined in what follows. First, ALBs are derived for each subsystem. The capacities are reallocated based on the ALB within the subsystem in a second step. In the third step, an ILB is determined as the minimum overall over all subsystems.

3.1. Generation of ALBs

Figure 2 depicts the decomposition of the flow line into subsystems as introduced in Weiss and Stolletz (2015). The isolated optimization of a subsystem results in lower bounds that are valid for groups of buffers in the original line but do not hold for individual buffers. We therefore refer to these bounds as ALBs (Weiss and Stolletz, 2015).
Each subsystem consists of \( i \) stations and is assumed to operate independently of the remaining stations of the line. Blocking and starvation that may occur in the original line, due to interactions between stations not included in the subsystem or the limited supply in front of the first station, are neglected. Consequently, the isolated optimization of a subsystem results in the same or less total buffer capacity than in the entire line (Weiss and Stolletz, 2015). The optimal buffer capacity of station \( m \) in the isolated subsystem \( l = 1, \ldots, M - i + 1 \) of size \( i \) is denoted by \( b_{m,1,i}^* \). The allocated total buffer capacity in the subsystem, \( \sum_{m=1}^{l+i-2} b_{m,i} \), is a lower bound for the capacities of the respective buffers in the original line, \( \sum_{m=1}^{l+i-2} X_m \) (Weiss and Stolletz, 2015); see Table 1 for the notation.

### Table 1. Notation for the calculation of lower bounds.

| Indices: | Parameters: | Real-valued decision variables: | Integer decision variables: |
|----------|-------------|---------------------------------|-----------------------------|
| \( m = 1, \ldots, M \) | \( T^* \) | \( E[TH(X_1, \ldots, X_{M-1})] \) | \( X_m \) |
| \( l = 2, \ldots, M - 1 \) | \( B_m \) | | \( b_{m,i} \) |
| \( l = 1, \ldots, M - i + 1 \) | | | \( b_{m,1,i} \) |
| Subsystems of size \( i \) | Goalt throughput | Expected throughput obtained with buffer allocation \( X_1, \ldots, X_{M-1} \) | Buffer capacity behind station \( m \) |
| \( \sum_{m=1}^{l+i-2} b_{m,i} \) | Maximum capacity of the buffer behind station \( m \) | | Buffer capacity behind station \( m \) in subsystem \( l \) of size \( \sum_{m=1}^{l+i-2} b_{m,1,i} \) |

#### 3.2. Reallocation of buffer capacities within a subsystem

We calculate the ILB for a buffer \( m' \) in subsystem \( l \) of size \( i \) using the ALB \( \sum_{m=1}^{l+i-2} b_{m,1,i} \). The idea is to reallocate the buffer capacities of the ALB such that the capacity \( b_{m',1,i}^* \) for buffer \( m' \) is minimized under consideration of the throughput constraint (see the mathematical program (2)):

\[
b_{m',1,i}^* = \min_{m} b_{m',1,i}^*
\]

\[
\sum_{m=1}^{l+i-2} b_{m,1,i}^* \geq \sum_{m=1}^{l+i-2} b_{m,1,i}
\]

\[
E\left[T^* \left( b_{m,1,i}, \ldots, b_{m',1,i-2,1,i} \right) \right] \geq T^*
\]

\[
b_{m',1,i}^* \leq B_m, \quad \forall m = l, \ldots, l + i - 2
\]

\[
b_{m',1,i}^* \geq 0, \quad \text{integer, } \forall m.
\]

The objective function (2a) minimizes the capacity of buffer \( m' \) in the subsystem \( l \) to obtain an ILB for \( m' \). In Constraint (2b), it is ensured that the total number of buffer spaces of the candidate allocation is larger or equal to the ALB derived from the optimization of the subsystem \( l \). Additionally, the goal throughput has to be attained by the expected throughput of the subsystem obtained with allocation \( b_{m,1,i}^*, \ldots, b_{m',1,i-2,1,i}^* \). This is ensured by Constraint (2c). Constraints (2d) and (2e) ensure that the previously defined maximum buffer capacity, \( B_m \), is obeyed. In Constraints (2e), it is specified that the buffer capacities are non-negative and integer. Note that the sum of the ILBs, \( \sum_{m=1}^{l+i-2} b_{m,1,i}^* \), from subsystem \( l \) of size \( i \) is in general smaller than the corresponding ALB.

**Theorem 1.** \( b_{m',1,i}^* \) is an individual lower bound for the capacity of buffer \( m' \).
Proof. Weiss and Stolletz (2015) prove that $\sum_{m=1}^{l+i-2} b_{m,l,i}$ is an ALB for the total capacity of buffers $l, \ldots, l + i - 2$ with $l = 1, \ldots, M - i + 1$ and $i = 1, \ldots, M - 1$. Constraint (2b) holds because $\sum_{m=1}^{l+i-2} b_{m,l,i} < \sum_{m=1}^{l+i-2} b_{m,l,i}$ violates the ALBs. Constraints (2c) to (2e) formulate Equations (1b) to (1d) for the subsystems and therefore only exclude candidate allocations that are also excluded when calculating the ALBs using Equation (1). Consequently, the feasible region of the mathematical program (2) consists of all optimal allocations for subsystem $l$ of size $i$. The objective function (2a) minimizes the capacity of buffer $m'$. Thus, when solving the mathematical program (2), the result will be a feasible buffer allocation for the subsystem with minimum capacity of buffer $m'$; i.e., a lower bound for the capacity of buffer $m'$.

### 3.3. Derivation of minima

To obtain all ILBs from a subsystem $l$ of size $i$, the mathematical program must be solved for each buffer $m' = l, \ldots, l + i - 2$. Moreover, we calculate the ILBs resulting from different subsystems $l = 1, \ldots, l - 1$ of sizes $i = 3, \ldots, M - 1$, starting with $m' = 1, l = 1$, and $i = 3$. Consequently, several ILBs are obtained for each buffer $m'$.

As the different ILBs for a buffer $m'$ dominate each other, only the most restrictive ILB—i.e., the maximum value—is used for the optimization of the entire line. The constraints resulting from the ILBs,

$$\max_{l, i} b^{m*}_{m,l,i} \leq X_m \quad \forall m,\quad (3)$$

can also be used iteratively in the calculations of the ALBs and ILBs for all buffers of the subsystems $l + 1, l + 2, \ldots$ of size $i$ as well as for the calculations of larger subsystems $i + 1, \ldots, M - 1$.

In general, these bounds can be calculated with any buffer allocation algorithm, as they can be derived by simply optimizing different subsystems of the line. Additionally, such bounds can speed up different heuristic and exact solution approaches.

### 4. Rule-based local search algorithm

The Rule-Based Local Search (RBLS) algorithm solves the BAP under consideration of the pre-calculated lower bounds on the individual buffer capacities. Thereby, the RBLS algorithm iteratively applies a generative (see Section 4.1) and a sample-based evaluative method (see Section 4.2) as depicted in Fig. 3 to determine the optimal solution. The exchange of information on feasibility and optimality between generative and evaluative method is ensured by feasibility cuts and upper bounds.

This algorithm yields sample-exact buffer capacities, which converge to the exact optimum for sufficiently large samples.

#### 4.1. Generation of candidate allocations

Generating a new buffer candidate allocation is a non-trivial task, due to the complex relation between buffer spaces and throughput, which cannot be expressed in a closed form under general assumptions. We therefore develop an RBLS approach for the generation of candidate allocations.

The maximum buffer capacities, $B_1, \ldots, B_{M-1}$, are defined by the user and serve as a starting point for a solution. The generative method systematically generates new candidate allocations under consideration of the pre-calculated lower bounds and the results of the evaluation. Three cases can be distinguished.

(Case I) If the evaluation of the current candidate allocation results in a feasible throughput, a new candidate allocation is generated by reducing the capacity of one of the buffers by one. To decide which buffer capacity is to be reduced, a buffer selection criterion is used for the evaluation. Several criteria have been tested (see Section 5.1). The capacity of the buffer selected by the chosen selection criterion is reduced by one.

(Case II) If the evaluated allocation is infeasible, the evaluation of the remaining feasible allocation is used to determine another buffer, if available, based on the selection criterion. The respective buffer capacity of the last feasible allocation is reduced by one.

(Case III) If the evaluated allocation is infeasible and all its neighborhood candidates already have been evaluated, an artificial allocation with a total buffer capacity of the current upper bound — 1 is selected. We choose the allocation in the middle of our candidate allocation vector.

Cases (I) and (II) represent the local search of the algorithm. Case (III) represents a search in the global region to ensure that the algorithm finds the optimal solution and is not trapped in local optima.

Whenever all candidates have either been evaluated or excluded by bounds or cuts—i.e., no further candidate allocations exist—the last upper bound is equal to the optimal buffer allocation. If no upper bound was detected during the solution process, the problem is infeasible.

Despite the RBLSs, any other algorithm for the generation of candidate allocations can be used, if it ensures generating not only allocations in the neighborhood of current candidate allocations but also in the entire feasible region.
4.2. Sample-based evaluation and exchange of information

The candidate allocations are evaluated by a sampling algorithm with respect to the throughput, which is adapted from Chen and Chen (1993). Sample-based approaches model the flow of a large number of workpieces throughout the line. The random processing times, times to failure, and repair times are replaced by sampled effective processing times (i.e., processing times that already include down times due to failures), which are generated by Descriptive Sampling (Saliby, 1990). See Weiss and Stolletz (2015) for a detailed description of the sampling algorithm for the case of unlimited supply. We extend this algorithm to consider (r, S) and (s, q)-order policies modeling the supply of the first station.

If the throughput, $E[TH(X_1, \ldots, X_{M-1})]$, resulting from the evaluation is lower than the goal throughput, $TH^*$, the evaluated candidate allocation is infeasible. This candidate allocation, as well as all dominated allocations, is then excluded by feasibility cuts that are added to the RBLS. An allocation is dominated if all its buffer capacities are smaller or equal to the respective buffer capacities in the candidate allocation, see Weiss and Stolletz (2015). The lower bound on the total buffer capacity is (implicitly) increased if all candidates of a certain total number of buffer spaces are infeasible; i.e., all corresponding feasibility cuts have been generated.

If the candidate allocation is feasible, the upper bound on the total buffer capacity is updated to exclude all allocations with a higher or equal total number of buffer spaces.

5. Numerical Study

The algorithms are implemented in C++. Gurobi 5.0, with default settings and callbacks, is used to solve the mathematical programs described in Section 3. Callbacks are used to invoke the evaluation routine whenever Gurobi finds an incumbent solution. If the evaluation routine returns an infeasible throughput, the incumbent is rejected. The numerical study is performed on an Intel Core i7-3930K with 6x 3.2 GHz and 32 GB RAM.

To allow for the comparability of the numerical results, we use the test instances from Matta (2008), Helber et al. (2011), and Weiss and Stolletz (2015) in our numerical study. In all instance types, the total number of workpieces of a sample is set to $W = 250000$ and the warm-up phase is selected as $W_0 = 2000$. We generate 10 independent samples for each configuration. The detailed description of the test instances is given in the respective sections.

We first compare different selection criteria as part of the RBLS algorithm in Section 5.1. Section 5.2 investigates the performance of the RBLS algorithm and the ILBs. In Section 5.3, the impact of the order policies on the optimal buffer allocation is evaluated.

5.1. Impact of different buffer selection criteria

Table 2 shows 10 different selection criteria that were implemented within the RBLS algorithm. Both criteria from the literature and new criteria are tested.

| Table 2. Average computation times with different selection criteria (10 samples). |
|---------------------------------------------|
| **Criterion** | **Bottleneck last** | **Bottleneck middle** |
| **Av. comp. time (s)** | **Dev. from best (%)** | **Av. comp. time (s)** | **Dev. from best (%)** |
| Vergara and Kim (2009) | | | |
| Number of blocking events | 2616 | 1 | 1496 | 3 |
| Blocking time | 2581 | — | 1458 | — |
| Number of starvation events | 2696 | 5 | 1519 | 4 |
| Starvation time | 2597 | 1 | 1459 | — |
| Number of blocking and starvation events | 2621 | 2 | 1501 | 3 |
| Blocking and starvation time | 2727 | 6 | 1531 | 5 |
| New | | | |
| Net blocking time | 2909 | 6 | 1531 | 5 |
| Net starvation time | 2644 | 2 | 1486 | 2 |
| Li and Meerkov (2009) | | | |
| Equal protection criterion | 3375 | 31 | 1933 | 33 |
| Buffer half-full criterion | 2899 | 12 | 1696 | 16 |

Note. Lowest computation times are given in bold.

5.1.1. Criteria proposed by Vergara and Kim (2009)

Vergara and Kim (2009) propose several criteria based on blocking and starvation. The number of blocking events at a station $m$ and the blocking time of a station $m$ are related to the buffer behind station $m$. In contrast, starvation is caused by the buffer in front of station $m$. Therefore, the number of starvation events and the starvation time are related to the upstream buffer; i.e., the buffer behind station $m - 1$. The number of blocking and starvation events and the blocking and starvation time with respect to buffer $m$ are a combination of the above blocking and starvation criteria. In all cases, the capacity of the buffer with the lowest value of the criterion is decreased by one.

5.1.2. New criteria

Additionally, we test two criteria, which are also based on blocking and starvation times but have not been previously reported in the literature. The net blocking time criterion associated with the buffer behind station $m$ only considers blocking times that are caused by this buffer. This means that blocking times are only considered if station $m + 1$ is not blocked at the same time. The net starvation time criterion analogously considers only starvation times of station $m$ if station $m - 1$ is not starved at the same time.

5.1.3. Criteria proposed by Li and Meerkov (2009)

The equal protection criterion described in Li and Meerkov (2009) is based on the observation that buffer allocations with equal protection of station $m$ against blocking and starvation are good candidates for the optimal allocation. As a consequence, Li and Meerkov (2009) propose to calculate the indicator $E[WIP_m] - (X_{m+1} - E[WIP_{m+1}]) \forall m = 1, \ldots, M - 2$ where $E[WIP_m]$ is the expected Work-In-Process (WIP) in the buffer behind station $m$. This measures the balance of the expected WIP before station $m + 1$, $E[WIP_m]$, and the expected number of free buffer spaces behind station $m + 1$, $X_{m+1} - E[WIP_{m+1}]$. The idea is to enable a smooth flow by providing sufficient space behind the station for the expected amount of material in front of the station. Li and Meerkov (2009) describe the application of this criterion to the case of maximizing the throughput, subject to a constant total buffer capacity. We adapt the procedure for
the problem of capacity minimization subject to a throughput constraint as follows. Let $m$ be the station with the largest absolute value of the indicator. If the value of the indicator is positive, the capacity of the buffer behind station $m - 1$ is decreased by one, because the expected WIP in front of the station is too high compared with the expected number of free buffer spaces. Otherwise, the capacity of the buffer behind station $m$ is decreased by one, because the expected number of free buffer spaces is too high compared with the expected WIP in front of the station.

The buffer half full criterion arises from the observation that a full buffer protects best against starvation of the succeeding station, whereas an empty buffer protects best against blocking of the preceding station (Li and Meerkov, 2009). Thus, a buffer that is on average half full is a compromise between the two extreme cases.

To compare these criteria, we use a line with $M = 7$ stations, unlimited supply, and exponentially distributed processing times with a base processing rate of 7 units per time. The bottleneck is located either at the station in the middle of the line, or at the last station of the line, and has a processing rate of 6 units per time. The capacity of each buffer is limited to $B_m = 20$. The goal throughput is set to 5.776.

Table 2 shows the average computation times (resulting from 10 different samples) of the RBLS algorithm in combination with the different criteria (second and fourth columns). Note that the given computation times are generated with the RBLS algorithm in combination with ILBs. We have chosen the lowest computation times (bold) as the reference values for the calculations of the deviations in columns 3 and 5 for each type of instance. The lowest computation times for both bottleneck locations are obtained by the blocking time criterion. For a bottleneck in the middle of the line, the starvation time criterion results in the same average computation times. All other criteria containing blocking or starvation times also result in low computation times with only 1–6% deviation compared with the blocking time criterion. Consequently, the RBLS algorithm can be combined with any of these criteria. In contrast, the buffer half full criterion and the equal protection criterion result in a rather poor performance. Similar results were obtained for further numerical experiments with different distributions, number of stations, and upper bounds on the individual buffer capacities. In the following experiments, we apply the blocking time criterion of Vergara and Kim (2009). However, the superiority of certain selection criteria may depend on the structure of the instance chosen.

### 5.2. Impact of individual bounds and the RBLS algorithm

We first analyze the RBLS algorithm (based on the blocking time criterion) for unlimited supply. This allows for a comparison with the results of the Benders Decomposition in Weiss and Stolletz (2015). Both optimization algorithms are executed with ALBs (proposed by Weiss and Stolletz (2015)) and the new ILBs (as developed in Section 3). The experiments are based on the numerical studies of Matta (2008), Helber et al. (2011), and Weiss and Stolletz (2015). All instances investigate flow lines with $M = 7$ stations. The capacity of each buffer is limited to $B_m = 20$. The bottleneck is located either at the station in the middle of the line or at the last station of the line. We test instances with Erlang-1, Cox-2, and exponentially distributed effective processing times. To generate exponentially distributed effective processing times, the inverse transformation method is applied. The distribution parameters for Erlang-1 and Cox-2 are calculated using the formulas for mean and Squared Coefficient of Variation (SCV) in Bolch et al. (2006). These parameters are then used to generate the durations of the exponentially distributed phases. Finally, the effective processing times are calculated from the phase durations. Table 3 shows the parameters that change for the different distributions. The SCV measures the variability of the processing times and is calculated as the squared ratio of standard deviation to the mean.

Table 4 presents the average computation times resulting from 10 different samples for each of the different types of instances. The first three columns define the instance type. Column 4 contains the average computation time of the Benders Decomposition with ALBs as proposed in Weiss and Stolletz (2015). The fifth column depicts the results of the Benders Decomposition extended by the ILBs. The sixth and seventh columns consider the average computation times of the RBLS algorithm with ALBs and with ILBs, respectively. The eighth column shows the proportion of evaluated allocations of

---

**Table 3.** Parameter settings of the test cases.

| Processing time distribution | Erlang-1 | Cox-2 | Exponential |
|-----------------------------|---------|-------|-------------|
| Base processing rate        | 0.5     | 0.5   | 7.0         |
| Processing rate of bottleneck| 0.45    | 0.45  | 6.0         |
| Goal throughput $T^H$       | 0.405   | 0.405 | 5.776       |
| SCV                         | 0.25;0.5| 1;0;2.0| 1.0         |

**Table 4.** Performance comparison of the solution methods (average of 10 samples per test case).

| Distribution | SCV | Bottle-neck | Benders decomposition | RBLS algorithm | % eval. |
|--------------|-----|-------------|-----------------------|----------------|---------|
|              |     |             | ALB | ILB | ALB | ILB |         |
| Exponential  | 1.0 | Middle      | 100 | 62  | 45  | 24  | 0.67    |
| Exponential  | 1.0 | Last        | 125 | 76  | 81  | 43  | 1.11    |
| Erlang-4     | 0.25| Middle      | <5  | <5  | <5  | <5  | 0.03    |
| Erlang-4     | 0.25| Last        | <5  | <5  | <5  | <5  | 0.03    |
| Erlang-2     | 0.5 | Middle      | <5  | <5  | <5  | <5  | 0.03    |
| Erlang-2     | 0.5 | Last        | <5  | <5  | 6   | <5  | 0.04    |
| Cox-2        | 1.0 | Middle      | 30  | 17  | 16  | 10  | 0.22    |
| Cox-2        | 1.0 | Last        | 40  | 22  | 20  | 11  | 0.26    |
| Cox-2        | 2.0 | Middle      | 101 | 74  | 29  | 23  | 0.65    |
| Cox-2        | 2.0 | Last        | 775 | 375 | 104 | 76  | 2.02    |
the RBLS algorithm with ILBs from a total of $21^6 = 4084$ 101 possible allocations.

The results show that ILBs significantly improve the computation times of both the Benders Decomposition and the RBLS algorithm. Both algorithms solve the instances with Erlang-\(k\) distribution within a few minutes. For instances with exponentially or Cox-2-distributed processing times, a reduction of more than 65\% in the computation time is achieved. The most difficult types of instances are those with Cox-2-distributed processing times, an SCV of 2.0, and a bottleneck at the end of the line. The Benders Decomposition with ALBs takes on average 775 minutes to solve one instance of this type. With ILBs this time is reduced to 375 minutes. The RBLS algorithm with ALBs takes on average 104 minutes. This can be reduced to 76 minutes when applying ILBs.

The comparison of the computation times of the Benders Decomposition and the RBLS algorithm, independent of the considered lower bounds, reveals that the RBLS algorithm improves the computation times of the difficult instances. In particular, the instances with Cox-2-distributed processing times with an SCV of 2.0 and a bottleneck at the end are solved within 375 minutes with a Benders Decomposition (with ILBs), and this is reduced to 76 minutes by the RBLS algorithm. The computation times of instances with Erlang-\(k\)-distributed processing times remain roughly the same. The required computation time is only of the order of a few minutes and is therefore acceptable for both approaches. The number of evaluations during the execution of the RBLS algorithm with ILBs is very small, with a maximum of 2\% of the candidate allocations in the case of Cox-2-distributed processing times, an SCV of 2.0, and a bottleneck at the end of the line.

### 5.3. Impact of supply patterns

To investigate the impact of the order policies, we first optimize the total buffer capacity for a line supplied by a given \((s, q)\)-order policy with varying reorder points \(s\) and lead times \(T\). The order quantity is set to \(q = 200\), as numerical studies revealed that the order quantity only has a minor influence on the optimal buffer allocation. Second, the optimal total buffer capacity for a line supplied by a given \((r, S)\)-order policy with varying review intervals \(r\), order-up-to levels \(S\), and lead times \(T\) is investigated.

Finally, we present a study on the impact of the order policies on the computational performance.

In this section, we use an instance with exponentially distributed processing times and a bottleneck at the end of the line. This line was described in the previous experiment. The maximum capacity, \(B_m\), is set to the number of workpieces, \(B_m = 250\,000\). This corresponds to infinite buffers; i.e., we do not restrict the buffer capacities. Additional numerical experiments have been conducted with different distributions, processing times, and numbers of stations; however, they did not reveal further insights and are not reported here.

Figure 4 shows the optimal total buffer capacities for \((s, q)\)-policies with \(s = 943, \ldots, 1000\) and \(T = 165, \ldots, 167\). Thus, a total of 174 test cases is optimized. It can be observed that the required total buffer capacity increases exponentially with increasing \(T\) and decreasing \(s\), respectively. As the total buffer capacity cannot compensate for the lack of material induced by the pre-defined parameters of the order policy the BAP quickly becomes infeasible. Hence, there exist no feasible buffer allocations with respect to the goal throughput for certain values of \(s\) and \(T\), respectively. The figure includes no bars for these cases. Out of the 174 test cases, 23 cases are infeasible. The larger the lead time \(T\), the larger must be the reorder point \(s\) selected in order to attain the goal throughput. If \(s\) is chosen large or \(T\) is chosen small enough, this corresponds to an unlimited supply; i.e., the optimal buffer capacity for the case of limited supply converges to the optimal solution with unlimited supply for increasing \(s\) and decreasing \(T\) respectively.

Table 5 depicts the resulting buffer allocations for some selected \((s, q)\)-order policies. The first group (columns 1 to 9)

![Figure 4. Required total buffer capacity depending on the reorder point and the lead time for \(q = 200\).](image-url)

| \(s\) | \(T\) | \(\sum X_m\) | \(X_1\) | \(X_2\) | \(X_3\) | \(X_4\) | \(X_5\) | \(X_6\) |
|------|------|-------------|-------|-------|-------|-------|-------|-------|
| 1000 | 165  | 58          | 8     | 8     | 9     | 9     | 10    | 14    |
| 1000 | 166  | 58          | 8     | 8     | 9     | 9     | 10    | 14    |
| 1000 | 167  | 58          | 8     | 8     | 9     | 9     | 10    | 14    |
| 944  | 165  | 100         | 34    | 10    | 9     | 9     | 14    | 24    |
| 951  | 166  | 96          | 25    | 16    | 9     | 9     | 13    | 24    |
| 957  | 167  | 104         | 39    | 10    | 7     | 10    | 12    | 26    |
includes parameter choices that reflect unlimited supply. The allocations for unlimited supply ($s = 1000$) remain the same for varying lead time $T$. The second group (columns 10 to 18) corresponds to the lowest reorder points with a feasible solution for different lead times $T$. It can be seen that most of the capacity is located in front of the line and at the bottleneck; i.e., at the end of the line. The reason for adding buffer capacities in front of the line is that the lack of material induced by the limited supply is compensated by the additional buffer capacities. These capacities allow workpieces to enter the line, which subsequently triggers earlier replenishment. The reason for adding buffer capacities at the bottleneck is to prevent it from further starvation that is induced by the limited supply. Both the required capacity in front of the line and at the bottleneck increase rapidly with decreasing supply. In contrast, the capacities in the middle of the line remain almost the same.

Figures 5 and 6 show the optimal total buffer capacities for $(r, S)$ policies with $T = 165, \ldots, 167$ and $S = 1143, \ldots, 1200$ for $r = 35$ and $r = 40$, respectively. Thus, the buffer allocations for 348 test cases are optimized. The order quantity is set to $q = 1$; i.e., less than truck loads are allowed. Similar to the $(s, q)$ policy, the total buffer capacity increases with increasing $r$, increasing $T$, and decreasing $S$. This quickly leads to infeasibility. Out of the 348 test cases, 78 cases are infeasible. Moreover, if $r$ and $T$ are chosen small enough and $S$ is chosen large enough, unlimited supply is obtained.

Table 6 presents the resulting buffer allocations for selected $(r, S)$-order policies with $r = 35$. The first group (columns 1

| Table 6. Optimal buffer allocations for selected $(r, S)$-policies with $r = 35$. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $S$ | $T$ | $\sum X_m$ | $X_1$ | $X_2$ | $X_3$ | $X_4$ | $X_5$ | $X_6$ | $S$ | $T$ | $\sum X_m$ | $X_1$ | $X_2$ | $X_3$ | $X_4$ | $X_5$ | $X_6$ |
| 1200 | 165 | 58 | 8 | 8 | 9 | 9 | 10 | 14 | 1144 | 165 | 127 | 49 | 7 | 11 | 13 | 8 | 39 |
| 1200 | 165 | 58 | 8 | 8 | 9 | 9 | 10 | 14 | 1151 | 166 | 112 | 35 | 11 | 13 | 15 | 17 | 21 |
| 1200 | 167 | 58 | 7 | 9 | 9 | 9 | 10 | 14 | 1158 | 167 | 107 | 29 | 21 | 10 | 8 | 11 | 28 |
Figure 7. Computation times in relation to the total buffer capacity of the optimal allocation.

Table 7. Impact of neglecting limited supply in \((s, q)\)-order policies.

| \(s\) | \(T\) | Dev. from goal throughput (%) |
|-------|-------|------------------------------|
| 944   | 165   | -1.08                        |
| 951   | 166   | -1.03                        |
| 957   | 167   | -1.04                        |

Table 8. Impact of neglecting limited supply in \((r, S)\) order policies \((r = 35)\).

| \(S\) | \(T\) | Dev. from goal throughput (%) |
|-------|-------|------------------------------|
| 1144  | 165   | -1.16                        |
| 1151  | 166   | -1.08                        |
| 1158  | 167   | -1.04                        |

to 9) includes parameter choices that reflect unlimited supply. The allocations for unlimited supply \((S = 1200)\) remain almost the same for varying lead times \(T\). The second group (columns 10 to 18) corresponds to the lowest order-up-to-level \(S\) with a feasible solution for different lead times \(T\). It can be observed that the structure of the optimal allocations is similar to the case of the \((s, q)\)-order policies.

The impact of neglecting the limited supply is illustrated in Tables 7 and 8. The optimal allocations for the cases of unlimited supply from Tables 5 and 6 are evaluated under consideration of the order policies given in columns 1 and 2. Column 3 shows the deviation (in \%) of the resulting throughput from the goal throughput. It can be seen that this deviation is larger than 1\% in all cases. Hence, neglecting limited supply may result in buffer allocations that do not fulfill the pre-defined throughput goals. These results clearly demonstrate the need for solution approaches considering the supply patterns in front of the first station.

Figure 7 displays the impact of the order policies on the computation time. The computation times are obtained from an optimization run with maximum capacity \(B_m = 20\). Each point in the figure corresponds to a buffer optimization with given policy parameters. These policies are combined into groups with respect to the lead time \(T\). The reorder point \(s\) and the order-up-to level \(S\), respectively, determine the total buffer capacity on the x-axis; i.e., they are only implicitly considered in the figure. With changing parameters, the total buffer capacity increases or decreases as pointed out in the previous experiments. Therefore, Figure 7 shows the computation times in relation to different optimal total buffer capacities. The different curves of computation times induced by the \((s, q)\)-order policies and the \((r, S)\)-order policies do not differ significantly, which supports the observation that the policy parameters have a low impact on the computation time. Moreover, it can be observed that the computation time increases with the optimal total buffer capacity.

6. Conclusions and further research

In this article, we develop ILBs for the buffer capacities in flow lines. These bounds are derived by dividing the original system into subsystems and exploiting the fact that the subsystems are easier to solve. They can be applied in combination with any optimization algorithm for the BAP. Furthermore, we develop an RBLS algorithm that uses the bounds to optimally and efficiently solve the BAP with limited supply. This algorithm iteratively decreases the total buffer capacity based on the results of throughput evaluations. We compare several types of criteria to select the buffer whose capacity is decreased.

Our numerical study shows that the application of both the ILBs and the RBLS algorithm leads to substantial reductions in computation time. In addition, the numerical study reveals a significant impact of the limited supply on the optimal buffer capacity. Depending on the policy parameters, the optimal total buffer capacity increases exponentially. Thus, unless supplying the line with infinite material is not expensive, this work shows that the BAP cannot neglect the order policy governing the release of parts into the system.

Further research should extend this solution approach to take into account more complex systems, such as flow lines with closed loops or several product types. In addition, it is desirable to develop a model that allows for simultaneous optimization of the parameters for the order policy and the buffer capacities at any station of the line.
Notes on contributors

Sophie Weiss was awarded a Ph.D. from the University of Mannheim in 2015. She received her Diploma in Business Mathematics from the University of Cologne in 2010. Her main research interest is in the analysis and optimization of manufacturing systems.

Andrea Matta is an Associate Professor at the Department of Mechanical Engineering of the Politecnico di Milano. He graduated in industrial engineering at the Politecnico di Milano in 1998. Since then he has developed his teaching and research activities at the same institution. He was a Distinguished Professor at the School of Mechanical Engineering, Shanghai Jiao Tong University from 2014 to 2016. He has been Visiting Professor at the Ecole Centrale Paris (France), University of California at Berkeley, and Tongji University (China). He is responsible for the research area of Design and Management of Manufacturing Systems at MUSP (Laboratory for Machine Tools and Production Systems). His research interests include analysis, design, and management of manufacturing and health care systems. He is Editor in Chief of Flexible Services and Manufacturing Journal and is an editorial board member of OR Spectrum and IEEE Robotics and Automation Letters. He received the Shanghai One Thousand Talent and Eastern Scholar award in 2013.

Raik Stolletz studied mathematics, computer science, and business administration at the Technical University of Berlin. He received a Ph.D. from the Technical University of Clausthal in 2002 and the degree of Habilitation from the Leibnitz University of Hannover in 2009. In 2009–2010 he was an Associate Professor at the Technical University of Denmark. He currently is a full professor of production management at the Business School of the University of Mannheim. His research projects focus on quantitative decision support in the design and management of manufacturing and service systems. He has published articles in journals such as European Journal of Operational Research, Manufacturing & Service Operations Management, OR Spectrum, and Production and Operations Management.

References

Alden, J.M., Burns, L.D., Costy, T., Hutton, R.D., Jackson, C.A., Kim, D.S., Kohls, K.A., Owen, J.H., Turnquist, M.A. and Veen, D.J.V. (2006) General Motors increases its production throughput. Interfaces, 36(1), 6–25.

Alfieri, A. and Matta, A. (2012) Mathematical programming formulations for approximate simulation of multistage production systems. European Journal of Operational Research, 219(3), 773–783.

Alfieri, A., Matta, A. and Pastore, E. (2016) A column generation algorithm for the buffer allocation problem approximated by the time buffer concept. Manufacturing Modelling, Management, and Control, 49(12), 739–744.

Alfieri, A., Matta, A. and Pedrielli, G. (2015) Mathematical programming models for joint simulation-optimization applied to closed queueing networks. Annals of Operations Research, 231(1), 105–127.

Axsäter, S. and Rosling, K. (1993) Installation vs echelon stock policies for multilevel inventory control. Management Science, 39(10), 1274–1280.

Bolch, G., Greiner, S., de Meer, H. and Trivedi, K.S. (2006) Queueing Networks and Markov Chains: Modeling and Performance Evaluation with Computer Science Applications, 2nd ed. John Wiley & Sons, Inc., Hoboken, NJ.

Borzsonyi, M., Gershwin, S.B. and Suyematsu, C. (1998) Hewlett-Packard uses operations research to improve the design of a printer production line. Interfaces, 28(1), 24–36.

Buzacott, J.A. (1971) The role of inventory banks in flow-line production systems. International Journal of Production Research, 9(4), 425–436.

Chen, J.C. and Chen, L. (1993) A fast simulator for tandem queueing systems. Computers and Industrial Engineering, 24(2), 267–280.

Chiba, E. (2015) Heuristics for the buffer allocation problem with collision probability using computer simulation. Mathematical Problems in Engineering, 2015, 1–7.

Colledani, M., Ekvall, M., Lundholm, T., Moriggi, P., Polato, A. and Tolio, T. (2010) Analytical methods to support continuous improvements at Scania. International Journal of Production Research, 48(7), 1913–1945.

Costa, A., Alfieri, A., Matta, A. and Fichera, S. (2015) A parallel tabu search for solving the primal buffer allocation problem in serial production systems. Computers & Operations Research, 64, 97–112.

Dallery, Y. and Gershwin, S.B. (1992) Manufacturing flow line systems: A review of models and analytical results. Queuing Systems, 12(1), 3–94.

Demir, L., Tunali, S. and Eliiyi, D.T. (2014) The state of the art on buffer allocation problem: A comprehensive survey. Journal of Intelligent Manufacturing, 25(3), 371–392.

Gershwin, S.B. and Schor, J.E. (2000) Efficient algorithms for buffer space allocation. Annals of Operations Research, 93(1), 117–144.

Helber, S., Schimmelpfeng, K., Stolletz, R. and Lagershausen, S. (2011) Using linear programming to analyze and optimize stochastic flow lines. Annals of Operations Research, 182(1), 193–211.

Kolb, O. and Göttlich, S. (2015) A continuous buffer allocation model using stochastic processes. European Journal of Operational Research, 242(3), 865–874.

Köse, S.Y., Demir, L., Tunali, S. and Eliiyi, D.T. (2015) Capacity improvement using simulation optimization approaches: A case study in the thermotechnology industry. Engineering Optimization, 47(2), 149–164.

Köse, S.Y. and Kilinc, F. (2015) Hybrid approach for buffer allocation in open serial production lines. Computers & Operations Research, 60, 67–78.

Li, J. (2013) Continuous improvement at Toyota manufacturing plant: Applications of production systems engineering methods. International Journal of Production Research, 51(23–24), 7235–7249.

Li, J. and Meerkov, S.M. (2009) Production Systems Engineering, Springer, Boston, MA.

Li, L., Qian, Y.L., Du, K. and Yang, Y.M. (2016) Analysis of approximately balanced production lines. International Journal of Production Research, 54(3), 1–18.

Li, L., Qian, Y., Yang, Y.M. and Du, K. (2016) A fast algorithm for buffer allocation problem. International Journal of Production Research, 54(11), 3243–3255.

Liberopoulos, G., Takoumis, K. and Pandelis, D. (2015) Using shadow prices in a linear programming representation of kanban system dynamics to maximize system throughput, in Proceedings of the 10th international conference on Stochastic Models of Manufacturing and Service Operations, pp. 119–126. University of Thessaly Press, Volos, Greece.

Liberopoulos, G. and Tsarouhas, P.H. (2002) Systems analysis speeds up Chiptitis food-processing line. Interfaces, 32(3), 62–76.

Matta, A. (2008) Simulation optimization with mathematical programming representation of discrete event systems. In Proceedings of the 2008 Winter Simulation Conference, IEEE Press, Piscataway, NJ, pp. 1393–1400.

Matta, A., Pedrielli, G. and Alfieri, A. (2014) Event relationship graph lite: Event based modeling for simulation-optimization of control policies in discrete event systems, in Proceedings of the 2014 Winter Simulation Conference, IEEE Press, Piscataway, NJ, pp. 3983–3994.

Pedrielli, G., Alfieri, A. and Matta, A. (2015) Integrated simulation-optimisation of pull control systems. International Journal of Production Research, 53(14), 4317–4336.

Saliby, E. (1990) Descriptive sampling: A better approach to Monte Carlo simulation. The Journal of the Operational Research Society, 41(12), 1133–1142.

Shi, C. and Gershwin, S.B. (2016a) Lead time distribution of three-machine two-buffer lines with unreliable machines and finite buffers. Proceedings of the 10th international conference on Stochastic Models of Manufacturing and Service Operations, pp. 211–220. University of Thessaly Press, Volos, Greece.

Shi, C. and Gershwin, S.B. (2016b) A segmentation approach for solving buffer allocation problems in large production systems. International Journal of Production Research, 54(20), 6121–6141.

Silver, E.A., Pyke, D.F. and Peterson, R. (1998) Inventory Management and Production Planning and Scheduling, third edition, John Wiley: New York, NY.

Funding

This research was supported in part by the Julius-Paul-Stiegler-Gedächtnis-Stiftung.
Smith, J.M. (2016) Joint optimization of buffers and network population for closed finite queueing systems. *International Journal of Production Research, 54*(17), 5111–5135.

Smith, J.M. and Cruz, F.R.B. (2005) The buffer allocation problem for general finite buffer queueing networks. *IIE Transactions, 37*(4), 343–365.

Vergara, H.A. and Kim, D.S. (2009) A new method for the placement of buffers in serial production lines. *International Journal of Production Research, 47*(16), 4437–4456.

Wang, G., Shin, Y.W. and Moon, D.H. (2016) Comparison of three flow line layouts with unreliable machines and profit maximization. *Flexible Services and Manufacturing Journal, 28*(4), 669–693.

Weiss, S., Schwarz, J.A. and Stolletz, R. (2016) The buffer allocation problem: Formulations, solution approaches, and test instances. Advance online publication. https://doi.org/10.2139/ssrn.2843435

Weiss, S. and Stolletz, R. (2015) Buffer allocation in stochastic flow lines via sample-based optimization with initial bounds. *OR Spectrum, 37*(4), 869–902.