Proceedings of the 2017 Winter Simulation Conference
W. K. V. Chan, A. D'Ambrogio, G. Zacharewicz, N. Mustafee, G. Wainer, and E. Page, eds.

CONTROL AND DESIGN OF THE FILLET BATCHING PROCESS IN A POULTRY PROCESSING PLANT

Kay Peeters

Department of Mechanical Engineering
Eindhoven University of Technology
PO Box 513
Eindhoven, 5600 MB, THE NETHERLANDS

Tugce Martagan
Ivo Adan

Department of Industrial Engineering
Eindhoven University of Technology
PO Box 513
Eindhoven, 5600 MB, THE NETHERLANDS

Patrick Cruysen

Marel Poultry
Handelstraat 3
Boxmeer, 5831 AV, THE NETHERLANDS

ABSTRACT

In the poultry processing industry demand and supply are still growing in volume and diversity, which requires more processing capacity, flexibility and smarter control. This paper focuses on the fillet batching process. To minimize the giveaway of fixed-weight fillet batching the right choices on layout, buffer sizes, batch sizes and batch allocation policies are of great importance. We develop a simulation model to support such decisions on design and control. The model is used (i) to determine buffer and grader sizes, (ii) to optimize batch allocation in a dedicated layout, (iii) to compare a dedicated to a flexible layout and (iv) to assess the impact of smart allocation policies. In particular we find that significant reductions in giveaway can be achieved by employing so-called index policies in a flexible layout.

1 INTRODUCTION

Poultry processing is the industry where live broilers are processed to finished products. Specialized equipment is used by poultry processing plants to divide broilers into parts and package them. Naturally, the choice for producing chicken products depends on the demand from the market, which is made up out of supermarkets, fast food chains, restaurants and hotels amongst others. A processing plant using high-end equipment is typically capable of processing 200.000 broilers daily over a 16 hour period. Demand for end products ranges from whole broilers, to various types of wing, leg and fillet products and is steadily growing and expanding. Simultaneously, the supply of broilers is diversifying in terms of breeds, feed and living conditions. The research in this paper is conducted in close collaboration with Marel Poultry (Marel Poultry 2017c), which is a company that produces high-end poultry processing lines for the poultry processing industry. At Marel Poultry, the demand for more capacity, flexibility and control of these lines is still growing. As a result, Marel Poultry has expressed the need for tools to develop and quantify the benefits of new equipment, layouts and control strategies for poultry processing operations.

One of the products produced in a plant are batches of chicken breast fillets, which are a popular product in US and European markets. Each batch is a package containing fillets, which are produced in several steps using different types of equipment. Firstly, the production process starts with a continuous

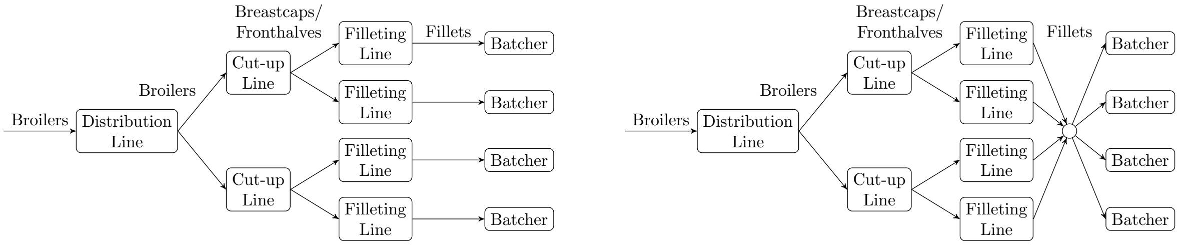
978-1-5386-3428-8/17/$31.00 ©2017 IEEE

flow of whole broilers (Figure 1a) supplied by the distribution process. Secondly, the broilers are cut up in the cut-up process to produce fronthalves (Figure 1b) or breastcaps (Figure 1c). Thirdly, the fronthalves or breastcaps are filleted in the filleting process to produce breast fillets (Figure 1d) and lastly the fillets are batched together in the batching process to produce batches of breast fillets.

(a) Broiler (b) Fronthalf (c) Breastcap (d) Breast fillet

Figure 1: Broiler products.

A schematic overview of the entire fillet batching process is shown in Figure 2a. The distribution line that supplies the flow of broilers has a fixed capacity and two cut-up lines are necessary to process all products. Similarly, two filleting lines are necessary to process all products coming from a cut-up line, and one batching machine is necessary to process the flow from a filleting line. In the current layout, the outflow from a filleting line is dedicated to a single batching machine. This layout will be referred to as the dedicated layout. However, Marel Poultry wants to explore the benefits of a more flexible concept in which all flows of fillets are merged and redistributed over the available batching machines. The schematic process layout of this flexible layout can be found in Figure 2b.



(a) Dedicated layout (current). (b) Flexible layout (concept).

Figure 2: Process overviews.

Batching machines can make two types of batches: fixed-weight and catch-weight batches. The former requires batches to have a minimum weight each, whereas the latter requires an average weight per batch over an entire production order. In this paper, we focus on the production of fixed-weight batches, where additional weight over the target weight, or give-away, is not paid for by customers. This give-away is the primary key performance indicator (KPI) in the fillet batching process.

The objective of this paper is to compare the performance of the dedicated layout to the flexible layout. In particular we aim to address design issues such as the size of buffers and graders and operational control issues such as the allocation of batch sizes in a dedicated layout and the dynamic allocation of fillets to batches. To this end we develop a simulation model for both process layouts and the model is created such that various algorithms for fillet allocation can be easily implemented and evaluated. Ultimately the performance of both layouts will be compared in terms of giveaway.

We summarize our contributions as follows. We study the real-time (on-line) variant of the bin-covering problem with a weight distribution based on customer data, and show that using smart algorithms and a flexible layout, a significant gain in performance can be achieved over dedicated layouts. To the best of our knowledge, we are the first to link the real-time bin-covering decisions with index policies, and consider an on-line problem with multiple batch sizes with realistic weights and limited available bins. Lastly, the paper provides insight to practitioners in designing and controlling their multi-machine batching operations.

The paper is organized as follows. In Section 2, the processes and control of a plant are described. The simulation model is outlined in Section 3. The results of the simulation are provided in Section 4 and our conclusions and recommendations for future work are summarized in Section 5.

2 FILLET BATCHING PROCESS

Product weight is an important factor in processing poultry and is discussed in Section 2.1. Next, the allocation decisions between processing lines will be considered. Lastly the production steps, and the processes leading up to them will be described in more detail.

2.1 Product Weight

Product weight is an important variable in processing lines for two reasons discussed below. Firstly, the weights placed in a batch determine the giveaway. Secondly, because processing lines are mechanically limited they can only handle a certain range of weight. Therefore, they are calibrated intelligently so that all lines together cover the entire weight range. This is necessary, because if a product is too light for a line, not all of the desired meat is cut off a carcass, which leads to lower yield. On the other hand, if a product is too heavy for a line, bone can be cut along with the meat leading to low quality or reworked products. Both situations lead to lost revenue. In addition to calibrated lines, buffers with a fixed size are used to hold excess supply. Furthermore, *Floating Weight Ranges (FWR)* are used in the allocation from distribution to cut-up, and cut-up to filleting lines as shown in Figure 3. An FWR is centered around the weights which split the product flow and work as follows. Products below a FWR will be allocated to the line with the lighter calibration, whereas products above this range will be allocated to the line with the heavier calibration. Products within the floating-weight range are allocated to the line with the emptiest buffer. This behavior is illustrated in Figure 3. Here, a Normal distribution of weights is used for illustration purposes, a hatched area indicates a FWR and open squares indicate buffers. It is assumed that a flow is balanced over its two downstream lines.

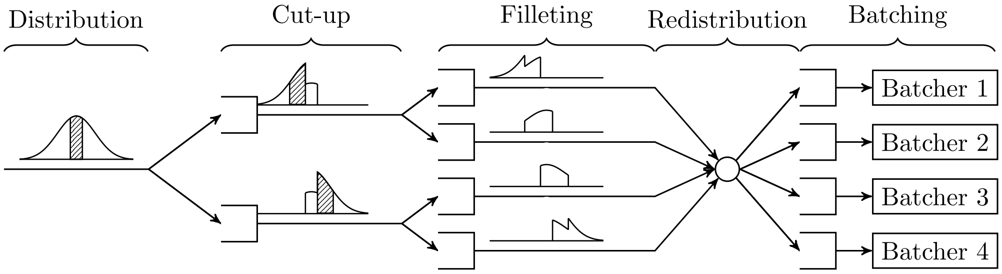


Figure 3: Example distribution of weight over cut-up and filleting lines.

The weight of broilers depends on a variety of factors. These include the breed of a broiler, what it has been fed, which season it is and if a flock has had a disease amongst others. Furthermore, there are natural differences between individual chickens. The result is that each weight range within a flock has a given frequency, resulting in a *probability distribution* of weight. A customer of Marel Poultry has provided us with information about the weight distribution of flocks they have processed, where three classes of flocks are distinguished: light, medium and heavy. Each broiler weight is allocated to a weight bin with 50 grams of size. The relative frequency of the combined broiler weights per weight bin of the light flock (dashed line) is shown in Figure 4. Using the sample mean and standard deviation of the empirical distribution, a Normal probability distribution is fitted (dotted line). One can observe that both distributions closely match, justifying the assumption that broiler weights are Normally distributed. It is noted that these weights only correspond to whole broiler weights, while the focus lies on breast fillets. A rule of thumb is that 25% of this weight is breast fillet, or 12.5% of the total broiler weight per fillet.

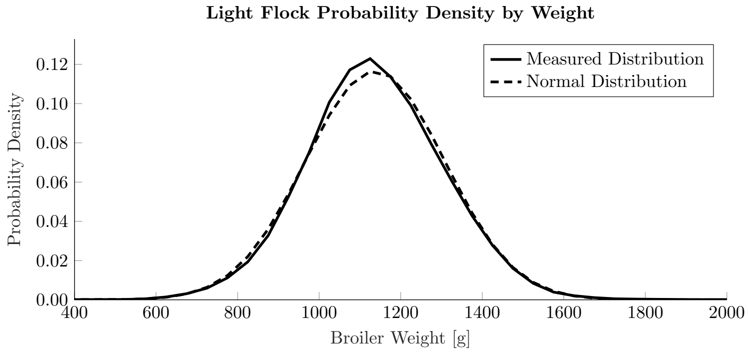


Figure 4: Weight distribution of light flock and fitted Normal distribution.

In conclusion we assume that the supply of broilers is Normally distributed, three weight distributions are considered, each fillet weighs 12.5% of the total broiler weight and each broiler yields two identical fillets. Lastly it is mentioned that a steady-state situation is assumed, where the weight distribution does not change over time, nor do the sizes of the batches that are being produced. This is a realistic assumption since flocks and orders are large and can take hours to process and produce.

2.2 Physical Processes

Flocks of broilers are delivered to a processing plant in trucks after which a flock is placed in the *primary process*. Firstly, the broilers are slaughtered. Secondly, the broilers are hanged in shackles by their feet, defeathered, eviscerated and cooled. After cooling down, broilers are moved to the *secondary process*. In this area, the chickens (Figure 1a) are cut up into different parts and processed to become the products one can find in supermarkets. The part that we will focus on are breast fillets (Figure 1d), which are produced by processing fronthalves (Figure 1b) or breastcaps (Figure 1c) depending on the equipment used.

After entering the secondary process, broilers are hanged in the distribution line that divides the broilers over two cut-up lines. This is the first step that is considered in the fillet batching process, as shown in Figure 2a. Within a cut-up line either fronthalves or breastcaps are produced, which are transported to filleting lines by conveyor. The filleting process uses Automated Modular Filleting lines (AMFs) (Marel Poultry 2017a) or Front Half Filleting lines (FHFs) (Marel Poultry 2017b) to process breastcaps or fronthalves respectively. However, the end result is the same: two fillets are obtained from each breastcap or fronthalf and placed on a conveyor, which transports them to a batching machine. The capacity of two filleting lines is necessary to process the entire flow of breastcaps or fronthalves coming from a cut-up line.

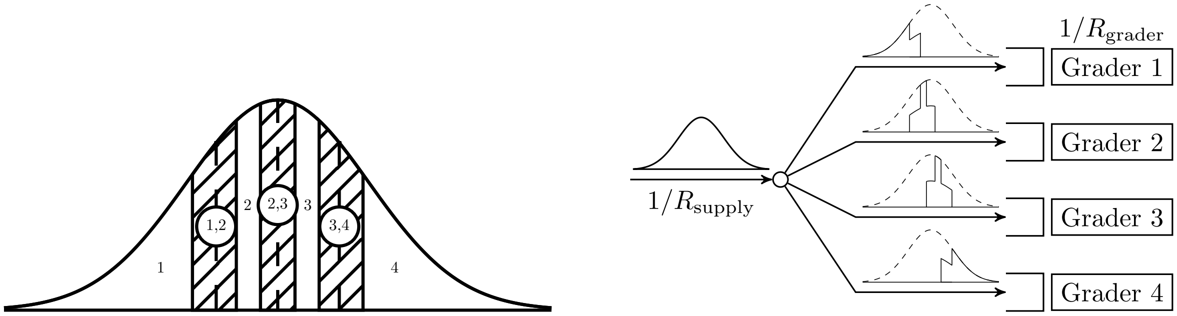
Several types of batching machines are used in practice, but our focus will be on *graders* as they are commonly used to batch fillets. Graders operate as follows: fillets arrive on a conveyor, are weighed and moved into available *bins* by flaps. Each bin contains a batch in progress. When a batch is completed, the bin containing the batch is emptied so the empty bin can be used again. Each batch is packaged and labeled before it is placed in inventory to wait for shipping. However, the scope of this paper is up and including the emptying of bins. The reader is referred to Figures 2a and 2b for a schematic overview of the process steps described in this section.

3 SIMULATION MODEL

In this section the implementation of the system into a simulation model is presented. The modeling of arrivals of products, the graders and the batching process, the allocation of products to graders and the allocation of products to bins are discussed. The software used to simulate the system is Matlab.

3.1 Product Arrivals

In Section 2 the usage of FWR and their effect on the allocation of weights in the dedicated layout was discussed. A visualization of these ranges for a given distribution is shown in Figure 5a. In the figure, the hatched areas indicate the FWR and the numbers indicate which filleting line a product from this would be processed by, with 1 the lightest and 4 the heaviest calibration. The vertical dashed lines indicate where the cumulative density of the distribution is 0.25, 0.5 and 0.75 respectively. This balances the load evenly among the four graders. The center FWR corresponds to the allocation from distribution to cut-up lines, and the outer FWR correspond to the allocation from cut-up to filleting lines.



(a) Illustration of floating weight ranges.

(b) Dedicated allocation weight distributions.

Figure 5: Illustration of floating weight ranges in a dedicated layout.

The arrival of products is modeled as follows for the dedicated layout. Fillets are generated in pairs from the weight distribution, where weights are generated from a light, medium or heavy Normal distribution $\mathcal{N}(\mu,\sigma)$ with mean μ and standard deviation σ. Their parameters are $\mathcal{N}(142.1,21.4)$, $\mathcal{N}(181.9,31.0)$ and $\mathcal{N}(232.3,40.7)$ respectively.

If the supply line has a capacity of R_{supply} products per second every $1/R_{\text{supply}}$ seconds a pair of fillets is generated. They are immediately placed in the buffer that corresponds to their weight if their weight falls outside a FWR. If their weight falls within a FWR, the pair is allocated to either buffer that correspond with the range with probability $1/2$. It is assumed that the buffers in the cut-up and filleting lines are large enough so that all products falling within a certain range will be allocated to the buffers associated with it. Furthermore, it is assumed that in the dedicated layout, the grader buffers have infinite size.

For a given distribution and FWR, all products falling within an FWR are allocated to either downstream line with probability $1/2$ on average. Using Figure 5a as a reference where a FWR of ± 0.075 of the distributions cumulative density function is used, the expected input per grader is illustrated in Figure 5b. Here the dashed lines correspond to the shape of the original distribution.

For the flexible layout, fillets are again generated in pairs from the weight distribution, but can be allocated individually and freely over the buffers of the graders. However, we want to use a finite buffer size here. If the allocation algorithm attempts to send many products to the same grader, the other graders may be starved and a limited buffer size naturally controls this behavior.

3.2 Graders and Batching

All graders allocate products in the same manner, have the same capacity, the same number of bins (8) and the same buffer size. The reasoning for the number of bins can be found in Section 4.1. The combined capacities of all graders is equal or greater to the arrival rate of products to ensure stability. As soon as there is a product in a buffer, the corresponding grader can allocate it to a bin immediately. In order to incorporate the grader capacity of R_{grader} products per second, the grader will wait $1/R_{\text{grader}}$ seconds before being able to allocate the next product. All bins within a grader have the same target weight B and are cleared immediately when this weight is reached or surpassed.

A grader makes use of an algorithm to decide in which bin to place each arriving fillet. It is continuously trying to minimize the giveaway of each bin produced. This type of problem is known as the *bin-covering problem*.

Definition 1 The *bin-covering problem* is defined as follows. For a given list $L = (a_1, \dots, a_n)$ of items (where a_k denotes the item size and k is the item index), with $a_k \in (0,1], \forall k \in \{1, \dots, n\}$, the goal is to pack all items into a *maximum* number of bins of size 1, such that each bin is at least filled to 1.

The bin-covering problem is considered *off-line* if all arriving items can be sorted and are known in advance, or *on-line* if the items arrive in some given order and must be allocated to a bin in that order. Next, a literature review is provided on the on- and off-line variants of the bin-covering problem, as well as index policies. The *off-line* variant of the bin-covering problem is reviewed first. Assmann et al. (1984) is the first to analyze this problem, and considers *Parametrized First-Fit Decreasing (FFD(r))*, where the parameter $r \in (1,2)$ is the modified bin size. After packing, bins below size 1 are combined. Their *Iterated Lowest-Fit Decreasing* algorithm maximizes the minimum bin level for a given set of items and bins. Csirik et al. (1991) modifies the Pairing algorithm from Knödel (1981) to obtain the *Pairing Heuristic*. Runarsson et al. (1996) use a genetic algorithm to solve the bin-covering problem.

On-line bin-covering has received less attention than the off-line variant. Assmann et al. (1984) considers the simple *Next-Fit (NF)* algorithm, which was adopted directly from its bin-packing counterpart. Arriving items are placed into a single bin until its target weight is reached, then closed, after which a new bin is opened. Csirik et al. (2001) extend the *Sum-of-Squares (SS)* bin-packing algorithm from Csirik et al. (1999) to bin-covering. More recent efforts include Asgeirsson and Stein (2006) and Asgeirsson and Stein (2009), who use Markov chains to model the bin-covering problem for a given distribution of items. Lastly we mention Ásgeirsson (2014) who introduces the *Prospect (PR)* algorithm, which uses information on the item distribution to estimate how easy it will be to fill a bin with small giveaway, as a function of the empty space left in it. We note that not all algorithms can be translated to a K-bounded variant, deal with multiple bin sizes at the same time, or achieve good performance when a subset of the item range $(0,1]$ is used.

Index policies, also known as *Gittins index* policies play an important role in the theory of Multi-armed bandit problems. The Gittins index introduced by Gittins (1979), gives an optimal policy for maximizing the expected discounted reward. An index is a direct measure of the expected reward that can be achieved from one machine, and choosing the maximum index at each period corresponds to the optimal policy. The PR algorithm is an example of an index policy, where the prospect (index) of a given bin gives a direct measure of the expected giveaway.

In our study, we show that certain bin-covering algorithms can be characterized as index policies. Additionally, most research focuses on the standardized bin-covering problem with a uniform weight distribution, a single batch size and unlimited bins. Instead, we address the bin-covering problem in a multi-batch size setting with product weights seen in practice and limited bins, for which there exist few results. Insight is provided into the effectiveness of two different bin-covering index policies and a benchmark policy. By comparing a dynamic and a dedicated layout, we show the added value of a dynamic layout, and demonstrate how different algorithms can make use of the added flexibility. Lastly, the effect of batch size and weight distribution is analyzed in a simulation study.

3.3 Index Policy

Let us now describe an index batching algorithm for the on-line K-bounded bin-covering problem that is applicable to graders. Consider a probability distribution that characterizes the weight of fillets. The minimum weight encountered is denoted by w_{min} and the maximum weight encountered by w_{max}. Weights can only take on discrete values and the entire set of weights encountered is denoted by $W = \{w_{min}, \dots, w_{max}\}$.

The probability that some weight is encountered is denoted by $p(w)$ for all $w \in W$. It is noted that $\sum_{w \in W} p(w) = 1$.

Clearly, when a batch is completed and its target weight B is reached or surpassed, the giveaway is the weight in the batch minus the target weight. We define an index $c(v, B)$ for each batch. The index indicates the expected give-away, given that its current content is v. Given that all full bins are closed instantaneously when they reach a weight of B or above, Equation (1) gives the giveaway of a completed bin.

$$c(v, B) = v - B, \quad v > B - 1. \tag{1}$$

Let w denote the random weight of an arriving item and consider a batch with index $c(v, B)$ that is not yet completed ($v < B$). The index for this batch can be computed by $c(v, B) = E(c(v + w, B))$ to obtain the expected giveaway given a random arrival. In fact, all indexes can be computed recursively for $v = B - 1, B - 2, \dots, 0$ by using Equation (2).

$$c(v, B) = E(c(v + w, B)) = \sum_{w = w_{min}}^{w_{max}} p(w) c(v + w, B), \quad \text{for } v = B - 1, B - 2, \dots, 0. \tag{2}$$

It is noted that $c(0, B)$ immediately yields the expected giveaway for a bin of size B given a *random allocation* of products. In fact, $c(0, B)$ corresponds directly with the expected giveaway if a NF algorithm were used (i.e. arbitrary allocation). The expected costs have now been defined for any weight in a bin. Suppose a grader is equipped with bins $K = \{1, 2, \dots\}$, and an item with weight $w \in W$ arrives at this grader. Furthermore, the weight in a bin is denoted by v_k for all $k \in K$. Using this information the expected giveaway for all bins after allocating w weight to it. This observation leads us to the *Index Policy (IP)* algorithm as defined by Algorithm 1. The expected giveaway can be calculated for any batch size and any item weight distribution.

Algorithm 1 The *Index Policy (IP)* for K-bounded on-line bin-covering consists of the following steps.

1. Using Equations (1) and (2), recursively calculate the indexes $c(v, B)$ for all possible weights in a bin v with target weight B. Go to step 2.
2. When an item with weight w arrives to a grader with K bins, that contain some weight v_k each, calculate the indexes for all bins $c(v_k + w, B)$. Go to step 3.
3. Allocate the item to the bin that yields the minimal index $c(v_k + w, B)$. Return to step 2.

As noted before, the PR algorithm can also be viewed as an index policy. The main difference with the above algorithm is that the index function $c(v, B)$ is replaced by a more advanced one.

3.4 Allocation to Graders

In the flexible layout it must be decided to which bin and to which grader to allocate an arriving fillet. Again the allocation should be such that it minimizes giveaway. Simple policies such as round robin or emptiest buffer allocation can be used, but are not expected to yield good performance as they do not utilize any information from the weight distribution or fullness of bins. Instead, we propose an extension of the IP algorithm, where multiple batch sizes are considered in parallel. This is the case when products can be allocated to graders that are producing different batch sizes. Let $G = \{1, 2, \dots\}$ denote the graders to which we can allocate an arriving product, B_g the batch size each grader $g \in G$ is producing, and $v_{g,k}$ the weight contained in bin k of grader g. Next, an *Index Policy for Multiple bin sizes (IPM)* is defined in Algorithm 2. Within the IPM algorithm fillets are allocated directly to a bin, rather than a grader.

Algorithm 2 The *Index Policy for Multiple bin sizes (IPM)* for K-bounded on-line bin-covering consists of the following steps.

1. Using Equations (1) and (2), recursively calculate the indexes $c(v, B_g)$ for all possible weights in a bin v for target weights B_g for all $g \in G$. Go to step 2.
2. When an item with weight w arrives to be allocated to a bin in any grader $g \in G$, that contain some weight $v_{g,k}$ each, calculate the indexes for all bins in all graders $c(v_{g,k} + w, B_g)$. Go to step 3.
3. Allocate the item to the bin that yields the minimal index $c(v_{g,k} + w, B_g)$. Return to step 2.

Both the IP and IPM algorithms can be used on-line and require the calculation of the indexes beforehand. In the order of $B \cdot (w_{max} - w_{min})$ calculations are needed to obtain all indexes for bin size B.

The described model and algorithm can be applied to more general settings. While four graders are considered, the problem can easily be adapted to an arbitrary number of graders with an arbitrary number of bins. Furthermore, even though the problem is described in the context of the poultry processing industry, it is common in the food processing industry as a whole, where fixed-weight packages play an increasingly dominant role. On the other hand, a bin-packing adaptation will find applications in other industries, such as the the postal & parcel industry, where sorting facilities allocate randomly sized packages to a fixed number of trucks.

4 RESULTS

In this section the research questions posed in the introduction will be answered using the simulation model introduced in Section 3. First, a suitable grader size will be determined. Secondly, for the dedicated layout it will be answered which batch size should be produced by which grader. Thirdly, a suitable buffer size for the flexible layout will be determined. Lastly, the performance of both layouts will be compared using the NF, IP and PR algorithms.

In this section, the performance of the system will be evaluated through four batch size combinations, or scenarios. Let $B = (B_1, B_2, B_3, B_4)$ denote the list of batch sizes that have to be produced on one of four graders. The four scenarios considered are given below in Table 1, where batch sizes are values typically encountered in practice. The first and third scenarios have a large spread in batch sizes with a low and high average weight respectively, whereas the second and fourth scenarios have a small spread with a small and large average weight respectively.

Table 1: Batch size combinations.

Scenario	B_1	B_2	B_3	B_4
1	400	600	800	1000
2	400	400	600	600
3	1000	1500	2000	2500
4	2000	2000	2500	2500

4.1 Grader Size

Graders come in different shapes and sizes. Graders with more bins are capable of creating more batches simultaneously so that the batching algorithm has more freedom of choice. In order to select a suitable grader size, the system is simulated to evaluate its performance. Scenario 1 is used in the simulation runs. In addition, the heavy weight distribution is used, the flexible layout is considered and the IP algorithm is used. 100.000 batches per simulation run are generated of which the results can be found in Table 2, where all digits shown are correct.

Table 2: Percentage giveaway for scenario 1 and the heavy distribution using IP.

Grader Size	1	2	4	6	8
Giveaway (%)	9.25	8.24	8.02	8.00	8.00

One can observe a strong decrease from 1 to 2 bins, another decrease from 2 to 4 bins and a small decrease from 4 to 6 bins. For the IP algorithm it would suffice to use 6 bins per grader. However, Ásgeirsson (2014) recommends using 8 bins to optimize the performance of the PR algorithm. Therefore, in order to obtain a fair comparison 8 bins per grader are used in all following simulation runs.

4.2 Order Allocation

When considering the dedicated layout, we have to determine which grader should produce which batch size to minimize giveaway. Let N_1, N_2, N_3 and N_4 denote the expected weight distributions on the four lines due to FWR (see Section 3). The indexes $c(0, B_i)$ for $i = 1, 2, 3, 4$ for each of these distributions immediately yield the expected giveaway if the NF algorithm were used. However, it is necessary to use simulation to compute expected giveaway values for other algorithms. Therefore, we obtain the expected giveaway for the IP and PR algorithms by means of simulation of a single grader. The PR algorithm will serve as a benchmark algorithm.

Next, we describe the procedure for scenario 1 and the light distribution. Each combination of batch size, algorithm and quarter of the weight distribution resulting from the application of FWR is simulated for 50.000 bins for the NF and IP algorithms and for 10.000 bins each for the PR algorithm. The giveaway as a percentage of total processed weight is given in Table 3. The reason that less bins have been generated for the PR algorithm is that the algorithm requires computationally-intensive tuning. We can observe that for most combinations the average giveaway decreases for larger batch sizes. The reason is that the item size relative to the batch size decreases. Additionally, the giveaway can be very sensitive to the distribution used. For example, the giveaway when using the PR algorithm on a batch size of 400 yields a giveaway ranging from 1.81% to 19.36%.

Table 3: Percentage giveaway for the dedicated layout using the light distribution in scenario 1.

Batch size	NF				IP				PR			
	N_1	N_2	N_3	N_4	N_1	N_2	N_3	N_4	N_1	N_2	N_3	N_4
400	17.35	**14.08**	12.16	25.08	14.25	**6.30**	10.85	19.88	12.64	**1.81**	10.87	19.36
600	**10.73**	12.06	13.05	11.21	**4.13**	10.74	7.30	10.02	3.38	10.70	**1.84**	10.02
800	6.33	6.94	12.10	**6.46**	3.09	3.58	9.99	**4.04**	**3.15**	1.17	6.48	4.05
1000	6.01	7.55	**4.91**	8.44	3.00	6.04	**4.47**	4.21	0.60	3.77	4.50	**0.68**

Next, the batch size must be allocated such that the average giveaway is minimized. This turns out to be an *assignment problem.*

Definition 2 The *assignment problem* is defined as follows. Suppose there are n workers that can perform one of n different tasks. Let $C(i, j)$ denote the cost of worker i performing task j. The goal is to allocate workers so that the total cost of performing all tasks is minimized.

In our case the workers are the graders, and the tasks are processing the four dedicated weight distributions. This problem can be solved by brute-force search, or by using the *Munkres* algorithm developed by Munkres (1957) for large problem instances. The optimal ordering of batches produced from distributions $[N_1, N_2, N_3, N_4]$ are $[600, 400, 1000, 800]$ for NF, $[600, 400, 1000, 800]$ for IP and $[800, 400, 600, 1000]$ for PR. These values have can be found in bold in Table 3. This procedure can be applied to any distribution and any combination of batches.

4.3 Flexible Layout Buffer Size

In the flexible layout, buffers can be used to hold overflow to each grader. By simulating the flexible layout with different buffer sizes, a suitable buffer size can be determined. Scenario 1, the light distribution and the IP algorithm are used and 100.000 batches are completed per simulation run. A grader size of 8 bins is used (see Section 4.1). The results are shown in Table 4, where all shown digits are correct.

Table 4: Percentage giveaway for scenario 1 and the light distribution using IP.

Buffer Size	1	2	3	50
Giveaway (%)	4.64	4.57	4.56	4.56

One can observe that the influence of the buffer size on the total percentage giveaway is minimal. There is a slight decrease from a buffer size of 1 to 2, from 2 to 3 and from 3 onward the giveaway remains constant. Therefore, in the remainder of the paper a buffer size of 3 is used. We note that a buffer size of 1 is identical to not using a buffer, and depending on the costs of including such a buffer it may be more economical to exclude a buffer entirely.

4.4 Layout Comparison

In Section 3 the IPM algorithm has been introduced, which can be used to allocate fillets directly to bins, rather than graders in a flexible layout. In a similar way the PR algorithm can be employed and is used as a benchmark algorithm. A simple allocation alternative here is round robin for allocating products to graders, and NF within the grader.

In order to assess the impact of using a smart allocation strategy using the dedicated layout, the NF, IP and PR algorithms are used in combination with the order allocation procedure described in Section 4.2. For each algorithm and layout, the performance is evaluated for the 4 scenarios from Table 1 in combination with the light (L), medium (M) and heavy (H) weight distributions. The performance measure used is the percentage of giveaway of the total weight processed. The simulation results can be found in Tables 5 and 6, where the average percentage giveaway over all scenarios and distributions per algorithm can be found at the top of each table.

Table 5: Percentage giveaway of total weight per scenario and algorithm using the dedicated layout.

Scenario	NF (8.28)			IP (5.61)			PR (4.38)		
	L	M	H	L	M	H	L	M	H
1	7.93	9.91	12.02	4.81	6.71	7.36	2.51	6.66	8.48
2	10.72	13.70	18.44	8.46	10.64	15.65	7.24	9.67	14.48
3	3.59	4.38	6.72	2.05	2.47	3.16	0.23	1.26	1.23
4	2.81	3.75	5.27	1.50	2.08	2.41	0.12	0.26	0.46

Table 6: Percentage giveaway of total weight per scenario and algorithm using the flexible layout.

Scenario	NF (9.65)			IPM (4.92)			PR (2.36)		
	L	M	H	L	M	H	L	M	H
1	9.86	13.82	14.89	4.56	6.16	7.93	1.65	2.02	4.98
2	12.33	17.94	17.53	5.63	7.96	12.68	2.08	4.26	11.96
3	4.39	5.64	7.17	2.10	2.70	3.38	0.15	0.29	0.54
4	3.15	4.00	5.08	1.52	1.94	2.45	0.06	0.12	0.20

Comparing the average performance of the flexible layout to the average performance the dedicated layout per algorithm yields the following. The NF algorithm sees an overall giveaway increase of 16.69% (8.27% to 9.65%), the IP (IPM) algorithm sees a giveaway decrease of 12.32 % (5.61% to 4.92%) and the PR algorithm sees a giveaway decrease of 46.18 % (4.38% to 2.36%). We conclude that the NF algorithm indeed yields the worst performance and even sees a performance decrease in the flexible layout, which may be due to the increased variability of the inflow to each grader. However, the more sophisticated IP and PR algorithms can both utilize the increased flexibility to reduce giveaway. The PR algorithm especially sees a dramatic overall performance improvement.

5 CONCLUSION

The aim of this study has been to compare a dedicated and a flexible layout of a fillet batching operation in a poultry processing plant. To achieve this goal, a simulation study has been performed to evaluate the performance of both layouts using the NF, IP and PR algorithms and three fillet weight distributions.

Firstly, the size (number of bins) in the grader was considered. It was shown that 6 bins was sufficient to achieve minimal giveaway using the IP algorithm, but that 8 bins were chosen to provide a good comparison to the PR algorithm. Secondly, for the dedicated layout the allocation of batch sizes to graders was considered. Here, the weight distributions arriving to each graders are predetermined due to the use of floating weight ranges. Simulating all combinations of distributions and batch sizes yields a matrix of expected giveaway values. Solving this assignment problem yields the allocation of orders to graders that minimize the giveaway. Thirdly, we looked at the buffer sizes in the flexible layout. Simulating the system showed that a buffer size of 3 was sufficient to minimize the giveaway in our production setting. Lastly, using the obtained buffer sizes for the flexible layout, derived grader sizes and batch size allocation procedure for the dedicated layout both layouts were compared. It was shown that NF had the worst, IP the second best and PR the best performance. Furthermore, both IP and PR that use information about the arriving distributions, obtained an overall giveaway reduction of 12.31% and 46.18% respectively. As a result, we recommend using the PR algorithm in combination with the flexible layout to obtain the best performance.

Future work includes the development of (near) optimal dynamic allocation strategies, as this study has indicated the benefits of flexible layouts paired with smart allocation control. Another direction may be assignment policies that automatically adapt to (learn) changing weight distributions. Lastly, the considered problem may be expanded to include deadlines of orders, order scheduling, set-up times when switching between batch sizes and operator availability.

REFERENCES

Ásgeirsson, A. 2014. *On-line algorithms for bin-covering problems with known item distributions*. Ph. D. thesis, Georgia Institute of Technology.

Asgeirsson, E., and C. Stein. 2006. “Using markov chains to design algorithms for bounded-space on-line bin cover”. In *Proceedings of the Meeting on Algorithm Engineering & Expermiments*, 75–85. Society for Industrial and Applied Mathematics.

Asgeirsson, E. I., and C. Stein. 2009. “Bounded-space online bin cover”. *Journal of Scheduling* 12 (5): 461–474.

Assmann, S., D. S. Johnson, D. J. Kleitman, and J.-T. Leung. 1984. “On a dual version of the one-dimensional bin packing problem”. *Journal of algorithms* 5 (4): 502–525.

Csirik, J., J. Frenk, G. Galambos, and A. R. Kan. 1991. “Probabilistic analysis of algorithms for dual bin packing problems”. *Journal of Algorithms* 12 (2): 189–203.

Csirik, J., D. S. Johnson, and C. Kenyon. 2001. “Better approximation algorithms for bin covering”. In *Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms*, 557–566. Society for Industrial and Applied Mathematics.

Csirik, J., D. S. Johnson, C. Kenyon, P. W. Shor, and R. R. Weber. 1999. “A self organizing bin packing heuristic”. In *Workshop on Algorithm Engineering and Experimentation*, 250–269. Springer.

Gittins, J. C. 1979. “Bandit processes and dynamic allocation indices”. *Journal of the Royal Statistical Society. Series B (Methodological)*:148–177.

Knödel, W. 1981. “A bin packing algorithm with complexity O (n log n) and performance 1 in the stochastic limit”. In *International Symposium on Mathematical Foundations of Computer Science*, 369–378. Springer.

Munkres, J. 1957. “Algorithms for the assignment and transportation problems”. *Journal of the society for industrial and applied mathematics* 5 (1): 32–38.

Marel Poultry 2017a. "AMF-i breast cap filleting system". http://marel.com/poultry-processing/systems-and-equipment/broilers/deboning-skinning--trimming/breast-cap-filleting/breast-cap-filleting-systems/amf-i-breast-cap-filleting-system/1376?prdct=1&parent=1376.

Marel Poultry 2017b. "FHF-XB front half deboning system". http://marel.com/poultry-processing/systems-and-equipment/broilers/deboning-skinning--trimming/front-half-filleting/automatic-front-half-deboning/fhf-xb-front-half-deboning-system/422?prdct=1&parent=422.

Marel Poultry 2017c. "Marel Poultry". http://marel.com/poultry-processing.

Runarsson, T. P., M. T. Jonsson, and P. Jensson. 1996. "Dynamic dual bin packing using fuzzy objectives". In *Evolutionary Computation, 1996., Proceedings of IEEE International Conference on*, 219–222. IEEE.

AUTHOR BIOGRAPHIES

KAY PEETERS is a Ph.D. student of the Dynamics & Control research group at the Eindhoven University of Technology. He holds an MSc degree in mechanical engineering from the same university. His Ph.D. project is centered around the optimization of poultry processing plants, with a focus on control and design. His research interests include discrete-event simulation of manufacturing networks, Markov processes and queuing theory, in the context of applied research. His email address is k.peeters@tue.nl.

TUGCE MARTAGAN is an Assistant Professor and Marie S. Curie Research Fellow in the Department of Industrial Engineering at Eindhoven University of Technology. She received her Ph.D. in Industrial Engineering from the University of Wisconsin-Madison. Her research interests include stochastic modeling and optimization of manufacturing systems. Her email address is t.g.martagan@tue.nl.

IVO J. B. F. ADAN is a Professor of the Department of Industrial Engineering at the Eindhoven University of Technology. His current research interests are in the modeling and design of manufacturing systems, warehousing systems and transportation systems, and more specifically, in the analysis of multi-dimensional Markov processes and queuing models. His email address is i.j.b.f.adan@tue.nl.

PATRICK CRUYSEN is the Manager of Software Development and Smart Handling at Marel Poultry. He obtained a mechanical and software engineering degree from Fontys Hogescholen after which he worked as Manager of Software Development at Marel Poultry. He has been working at Marel Poultry for over 17 years. His email address is patrick.cruysen@marel.com.