APPLICATION OF EVOLUTIONARY ALGORITHMS FOR BAKERY PRODUCTION SCHEDULING

Florian T. Hecker[(a)], Bernd Hitzmann[(b)]

[(a), (b)]University of Hohenheim, Department of Process Analytics and Cereal Technology, Garbenstraße 23, 70599 Stuttgart, Germany

[(a)]Florian.Hecker@uni-hohenheim.de, [(b)]Bernd.Hitzmann@uni-hohenheim.de

ABSTRACT
The production in bakeries can be modelled as a no-wait permutation flow-shop based on the constraints and frame conditions given by the real production processes. A modified genetic algorithm, ant colony optimization and particle swarm optimization were used to analyse and optimize the production planning of a bakery production line that processes 40 products on 26 production stages. This setup leads to 8.2 x 10^{47} different possible schedules in a permutation flow-shop model and is thus not solvable in reasonable time with exact methods. The makespan of the production, an objective function of high economic interest, was analysed. In combination with the created model, the applied algorithms proved capable to provide optimized results for the scheduling operation within a restricted computational time of 15 min, reducing the makespan by up to 8.6 %.

Keywords: Bakery Production Planning; Modified Genetic Algorithm; Ant Colony Optimization; Particle Swarm Optimization; Flow-Shop Scheduling

1. INTRODUCTION

It is an economical and often also vital necessity throughout the different industry branches from producing to service companies to work on an optimal level concerning their efficiency or at least as close to this ideal as possible. Even more since the increased awareness of the finite nature of most of the today commonly used resources came up.

To reach a state of efficient production and sustainable or at least efficient use of resources is, besides the degree of efficiency of the utilized machines, for the most part a matter of optimal or at least efficient scheduling.

From its beginnings in the 1950's (Johnson, 1954; Jackson, 1955; Smith, 1956) flow-shop scheduling became an increasingly important tool for decision-making, mainly in the last 20 to 30 years, because of its potential to optimize manufacturing processes and by that creating significant savings for companies of all kinds of industries.

The main elements of which all flow-shop models are build are a set of *j* jobs, which have to be processed on a set of *m* machines. In most manufacturing facilities a job, e. g. a product, has to undergo a series of operations or processing steps related to specific machines. In many cases the jobs have to follow the same route through the production stages and the required machines are assumed to be set up in series. Such a manufacturing environment is then referred to as a flow-shop (FS). Some minor modifications or additional constraints of the aforementioned set up lead to special kinds of FSs. If the order in which the jobs are processed is the same for all stages of the FS, the FS is referred to as permutation flow-shop (PFS). In a more general machine set up that consists of *k* stages with a certain number of parallel machines *m* in each stage and *j* jobs that have to be processed on only one of these machines in each stage, the machine environment is referred to as flexible FS, compound FS or hybrid flow-shop (HFS) (Pinedo 2008).

Any kind of FS scheduling problem that includes a no-wait constraint for the jobs can be labelled as a no-wait flow-shop (NWFS) scheduling problem. This type of scheduling represents a very important application since it can be used for all kinds of time dependent industries like food or pharmaceutical production (Hall and Sriskandarayah 1996, Raaymakers and Hoogeveen 2000), as well as chemical, concrete or steel processing (Fondrevelle, Oulamara, Portmann, and Allahverdi 2009, Grabowski and Pempera 2000, Rajendran 1994).

A commonly faced problem in flow-shop scheduling is that it belongs to the class of NP-hard problems (Garey, Johnson, and Sethy 1976, Lenstra and Rinnooy Kan 1978), which means that the computational requirements for finding an optimal solution increase exponentially with the problem size. Thus large problems cannot be solved using exact methods, but a huge variety of heuristic and metaheuristic solution methods has been developed for flow-shop scheduling.

Due to the increasing importance of tackling complex multi-stage FS scheduling problems to model and solve real world scheduling problems to optimize manufacturing in numerous industrial branches, the range of approaches and solution methods employed is as various as the respective real world archetypes. Nevertheless there are many reviews available that summarize those approaches and methodologies, e. g. the works of Vignier et al. (1999) or Linn and Zhang (1999), Ruiz and Vázquez-Rodriguez (2010), Hejazi and Saghafian (2005), Gupta and Stafford (2006), Minella, Ruiz, and Ciavotta (2008) or Hall and Sriskandarajah (1996).

Three frequently used optimization algorithms for scheduling problems are genetic algorithms (GA), ant colony optimization (ACO) and particle swarm optimization (PSO). Due to the many examples of successful implementation of these methods it was decided to also apply them in this work.

Inspired by the Darwinian principles of evolution, the first GA was introduced by Holland (1975). Genetic and evolutionary algorithms use the same basic operations as in the reproduction and evolution of higher species, like inheritance of genes, mutation, selection and recombination. GAs and modified GAs are widely used for solving complex optimization problems (Fereidoonian and Mirzazadeh 2012, Gómez-Gasquet, Andrés, and Lario 2012, Phanden, Jain, and Verma 2012, Ventura and Yoon 2013, Zhang, Zhou, and Liu 2012, Ziaeifar, Tavakkoli-Moghaddam, and Pichka 2012).

Initially proposed by Dorigo in his Ph.D. thesis (Dorigo 1992) ACO adapts the mechanisms that help ants to find the shortest and thus optimal way between a food source and their formicary. Proven its capability to resolve the TSP (Dorigo and Gambardella 1997), one of the classical NP-hard problems, ACO is another nature inspired and frequently used algorithm to solve combinatorial optimization tasks. There is by now an almost inconceivable variety of applications using ACO, covering all kinds of scheduling, routing and optimization problems. The works of Dorigo, Di Caro and Gambardella (1999), Mullen et al. (2009), Chandra Mohan and Baskaran (2012) and of Tavares Neto and Godinho Filho (2013) provide an impressive overview of how versatile and successful ACO can be employed.

Kennedy and Eberhart (1995) invented PSO as an adaption of the movement and behaviour of bird flocks or fish schools on their search for a food source. Tasgetiren et al. (2007) were the first to tackle makespan and total flowtime minimization of a PFS using PSO. Minimizing C_{max} was also the objective function in the work of Lian, Gu and Jiao (2008) who used a novel PSO adapted to the discrete space of the PFS problem. Pan, Tasgetiren and Liang (2008) created a discrete PSO for a NWFS with makespan and total flowtime criterion and a new position update method as well as several speed-up methods were presented in their work. A *k*-stage NWFS was modelled with makespan as objective function for the scheduling of a polypropylene process by Liu, Gao and Pan (2011) and a hybrid PSO combined with SA was introduced to solve the scheduling problem.

The huge variety of applications of PSO is also presented in the works of Eslami et al. (2012) and Poli, Kennedy and Blackwell (2007).

Although there are these many examples where numerical modelling and optimization have been successfully applied in different industry branches, the baking industry in Germany yet provides no such efforts or applications. Even though this industry branch with its high diversity of products and time dependent production processes is as if predestined for the application of state-of-the-art scheduling methods. In the German baking industry, the production planning is almost completely based on the practical experience of the responsible employee(s) instead of the usage of mathematical methods. Regarding the high diversity of the product range in a common German bakery that includes around 50-100 different products and the high complexity of the scheduling task induced therein, the performance of bakeries is often sub-optimal.

The baking industry in Germany consists of approximately 14,000 producing companies, employs over 290,000 employees and reaches a business volume of almost 13.4 billion Euros per year (Zentralverband des Deutschen Bäckerhandwerks e. V. 2012). The increase of companies' efficiency in respect of energy consumption or staff allocation and man working hours therefore comprises high potential to decrease production costs.

2. MATERIAL AND METHODS

The model implementation, simulation and optimization were performed on a "lenovo ThinkPad R500" with an "Intel Core 2 Duo" 2.26 GHz processor, 2 GB RAM and Microsoft XP 2002 as operating software.

The modelling of a bakery production line with 40 products and 26 production stages, as well as the implementation of MGA, ACO and PSO and the optimization were programmed and performed with MATLAB R2012b (The MathWorks, Inc).

2.1. Modelling procedure

From the scheduling point of view the production in a bakery can be described as a hybrid flow-shop according to commonly used definitions (Pinedo, 2008; Ruiz & Vázquez-Rodriguez, 2010). The machine environment is called a hybrid flow-shop, if jobs have to be processed on only one machine *m* in a stage *k* or can completely bypass it, being the case in a bakery, as long as they are processed on at least one stage.

By considering the scheduling task in a bakery as a permutation flow-shop instead of a 'normal' hybrid flow-shop by adding the constraint, that the order in which the jobs *j* pass through the production is fixed and does not change between production stages (Pinedo, 2008), the number of possible product sequences can be reduced significantly. Although the real process in a bakery does not fulfil these requirements entirely, this model can be used and modified to match with the real production processes, where products can bypass earlier started products and the sequence of products on the first production stage determines all subsequent process tasks, due to the further down specified time dependence in bakery production. Doing so, the number of possible combinations is reduced from $(j!)^m$ to $j!$ and each possible schedule is a permutation of j (Pinedo 2008). Each of those permutations can be used to represent a sequence of products on the first production stage of the bakery which is crucial for sequencing the work flow. Still, high numbers of j, and thus $j!$ different schedules may lead to optimization problems unsolvable with exact methods in reasonable computational time.

Additionally the production in bakeries is commonly subject to a no-wait constraint due to inherent fermentative processes. The production of most baking goods requires proofing and the most commonly used proofing agent is yeast (*Saccharomyces cerevisiae*). The main characteristic of this process is the fermentative decomposition of glucose to CO_2, among other components. The retention of CO_2 produced by the yeast cells is given by the dough matrix surrounding the gas bubbles and leads to a desired volume increase. Up to a certain degree the dough matrix can withstand the structural stress induced by the, over time, increasing gas pressure, but after exceeding the maximum gas retention ability the dough matrix collapses. Due to these processes the production of such goods is not highly but strictly time sensitive from the point on, where the microorganisms get in contact with water and substrates under preferable conditions of temperature and humidity, as it happens in the dough production process. Cooling can be used to regulate or slow down the fermentation speed of yeast but is costly and sometimes accompanied with negative influence on the product quality.

2.1.1. Production modelling

Modelling the production site and ensuring the compliance of the no-wait constraint are done prior to the scheduling optimization.

All products in a bakery follow more or less the same way on consecutive stages through the production, meaning that a product does not return to an already passed stage. The common progression of these production stages is shown in Figure 1.

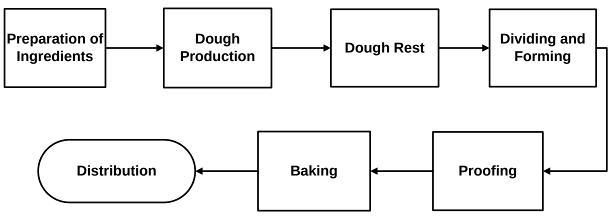

Figure 1: Basic Bakery Production Flow Model

Besides the definition of each products individual work flow through the production and the capacities of the employed machines, the processing times (*PTs*) of the products on each stage represent the most important information for modelling. *PT* also include process steps, where the product is not literally "processed" by means of being influenced by a worker or a machine, like the dough rest since this is a defined and desired waiting time. Table 1 shows an example containing the required processing information of products. The numbers in Table 1 represent the elements of a matrix $\underline{\boldsymbol{A}}$. The rows and columns of matrix $\underline{\boldsymbol{A}}$ represent the products and the production stages respectively, such that, for example $\underline{\boldsymbol{a}}_{2,3}$ would return the processing time of product 2 on stage 3, meaning in case of Table 1, that product "B" requires a forming time of 10 min. The recipe and the desired characteristics of the finished product determine the respective processing times.

Table 1: Example of bakery production data

Stage / Product	Dough production [min]	Dough rest [min]	Forming [min]	Proofing [min]	Baking [min]
A	5	0	5	35	25
B	4	20	10	35	55
C	8	30	15	50	30
D	6	25	25	40	60
E	9	0	8	55	40
F	10	10	12	35	35

Taking into account that some products do not have to be processed on all present stages (e. g. if a product needs no dough rest) and therefore might skip certain processing steps, a zero entry in matrix $\underline{\boldsymbol{A}}$ indicates that the specific product skips the respective stage and is not processed there.

Matrix $\underline{\boldsymbol{A}}$ is the basis to form a new matrix $\underline{\boldsymbol{B}}$. To do so the starting times (*STs*) of all products on all stages are calculated for the investigated sequence of products (which represents an individual in GA or a path of an ant in ACO or a particle in PSO for optimization). *ST* for the first product on stage 1 is "0" as this represents the start of the production shift and the *STs* on the successive stages are just a summation of the respective previous *PTs*.

Each investigated product sequence is scheduled following the procedure shown in Figure 2. The scheduling conditions include a reconciliation of the calculated *STs* of the currently processed product with the capacities and busy times of the involved machines, to make sure that the no-wait constraint is not violated and the sequence is valid for the following optimization procedure.

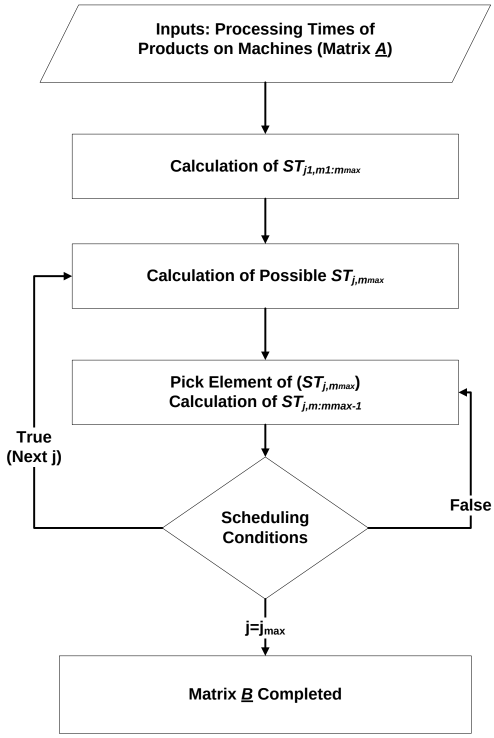

Figure 2: Flow chart of modelling algorithm without optimization procedure

Using the example data of Table 1 and defining that the processes "Dough rest" and "Proofing" are stages with unlimited capacity, matrices ***A*** and ***B*** will be obtained as shown in Figure 3, where ***A*** contains the processing times and ***B*** the starting times of products (row-wise) on the respective stages (column-wise), such that e. g. $\underline{\boldsymbol{b}}_{4,4}$ means that product "D" starts its proofing process 119 min after the shift start.

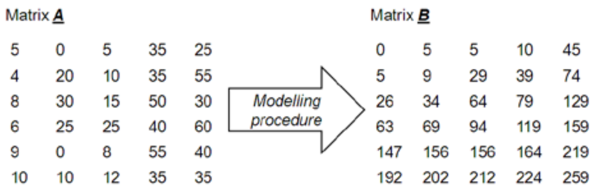

Figure 3: Example matrices A extracted from Table 1 and B with the same product order

To comply with the no-wait constraint, the processing start of some products has to be delayed, as can be seen in the third row of matrix ***B*** shown in Figure 3. The first machine would be available for product "C" at nine min after shift start, but starting the processing at this time would mean that the product has to be baked at 112 min after shift start. At this time the oven is blocked by product "B", which is baked from 74-129 min after shift start and product "C" would have to wait, thus violating the no-wait constraint.

A real bakery production in Germany including 40 different products and 26 different production stages was investigated in this study. Figure 4 shows the corresponding production line model. Modelling this production line as a permutation flow-shop leads to a total of 8.2 x 10^{47} (= 40!) different possible schedules.

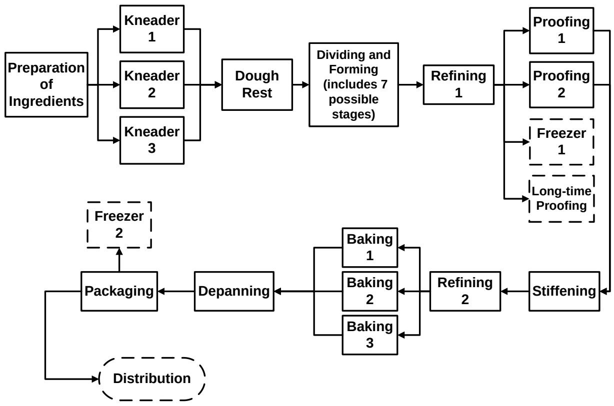

Figure 4: Model of the investigated production

The common way of a product would be to exit the production line as a baked good via the "Distribution" process. Nevertheless some products can exit the production e. g. without passing the baking process due to long-time proofing or being distributed as frozen dough pieces and thus do not influence the production after their specific final process step. The model therefore contains different exit points, shown as shapes with dashed lines in Figure 4. Any product following a long-time proofing procedure will be distributed or baked after the current work shift has finished and therefore exits at the "Long-time Proofing". And finally products that are distributed frozen have to be stored in a freezer and thus exit the model at the "Freezer 1" or "Freezer 2" stage.

The capacities of the employed machines or stages have a crucial importance for the scheduling process. The production stages can roughly be grouped into stages with limited capacity and stages with (practically regarded) unlimited capacity. The latter can handle any amount of different products without blocking under common production conditions. In the modelled bakery shown in Figure 4 "Dough Rest", "Proofing 1", "Proofing 2", "Long-time Proofing", "Stiffening", "Freezer 1", "Freezer 2" and "Depanning" belong to the group of stages with unlimited capacity. "Baking 1" has a limited capacity of two and "Baking 2" of six, meaning that these stages can process two or six products (or product batches) simultaneously. All other stages of the model have a limited capacity of one and can process only one job at a time.

2.2. Optimization procedures

As a quality measure of the analysed product sequence at least one cost or objective function to be optimized must be defined prior to the actual optimization process. The minimization of the makespan of the production was chosen as objective function in this work.

The makespan (C_{max}) represents the required time to complete a defined production goal and equals the highest end time (*ET*) of the products and thus can be easily retrieved from matrix $\underline{\boldsymbol{B}}$ and matrix $\underline{\boldsymbol{A}}$ by summation of the start of the last production step of the last product (e. g. $\underline{\boldsymbol{b}}_{6,5}$ in Figure 3) and the corresponding processing time (e. g. $\underline{\boldsymbol{a}}_{6,5}$ in Figure 3). For the example shown in Figure 3 C_{max} would thus be 294 min (259 min + 35 min).

The computational time was restricted to a maximum of 15 minutes to make the optimization procedure feasible for usage prior to a work shift in a real production environment. To match this constraint it was decided in the MGA to limit the number of individuals to 100 and the number of generations to ten. Analogous in the ACO and PSO the number of ants/particles was limited to ten and the number of iterations to 100, respectively.

2.2.1. Modified Genetic Algorithm

The classical GA is primarily not considered to be applied to combinatorial scheduling problems. Therefore the GA used in this work was modified to make it suitable to solve the combinatorial sequencing problem. The conducted modifications follow the ideas of the New Genetic Algorithm (NGA) introduced by Ventura and Yoon (2013) and of partially matched crossover (PMX) by Goldberg and Lingle (1985).

The first step consists of the calculation of the objective function values of all individuals of the initial population and their ordering according to the respective results. Afterwards the best 50 % of the population are selected for the mating and reproduction process. The mating in this MGA simply consists of coupling the selected individuals pair wise, such that the individual with the best objective function value is mated with the second best, the third best is mated with the fourth best and so on. As next step two offspring are created by each mated pair.

The creation of these offspring is performed by a PMX operation. In contrast to the NGA (Ventura and Yoon 2013) four crossover regions were defined such that the jobs on positions 1-5, 11-15, 21-25 and 31-35 in the parent sequences were exchanged as well as the respective jobs included in these sections, to ensure that the offspring sequences represent permutations without job duplications.

After the production of the offspring by PMX, mutation can occur with mutation probability of Mp = 0.1. If mutation occurs, two jobs in the sequence are chosen at random with a uniform distribution and change their respective places. The progress of offspring creation in the employed MGA is illustrated in Figure 5, showing the recombination of the two parent sequences P1 and P2 to create one example offspring.

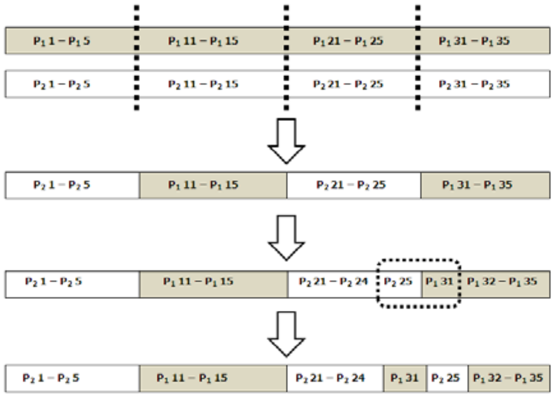

Figure 5: Creation of Offspring in MGA

The four crossover regions are separated by the three vertical dashed lines. After creating an offspring with recombined ‘genes’ mutation occurs (indicated by the dotted frame), causing two products to change their respective positions and creating a mutated offspring.

2.2.2. Ant Colony Optimization

Like GA, ACO is a nature inspired iterative optimization algorithm. In this case the activities and mechanisms observable in sedentary ant colonies are adapted, not the processes that happen during gene transfer in the reproduction of higher organisms. It is well known in biology that such ants often find the shortest and thus optimal way between a food source and their formicary over a certain time span.

The mechanisms behind this effect are also known and based on the special way of communication between ants via pheromones, a special group of evaporative biochemical molecules. If a foraging and randomly wandering ant finds a food source it heads back to its formicary, laying down a pheromone trail. Other ants crossing such a trail will stop wandering randomly and follow this trail to the food and start bringing it to the nest while also laying down pheromones and thus reinforcing the scent. With time the shortest and most frequently used way will thus hold the highest pheromone concentration and attracts the highest number of ants. Pheromones are not persistent and once a food source is exhausted, the previously marked and reinforced pheromone trails are less and less used and the scent dissipates over time.

In ACO an artificial ant “moves” by applying a local stochastic decision and while moving builds a possible solution to the given optimization problem, e. g. a product sequence as in the presented work. Figure 6 shows the process of creating such a possible solution.

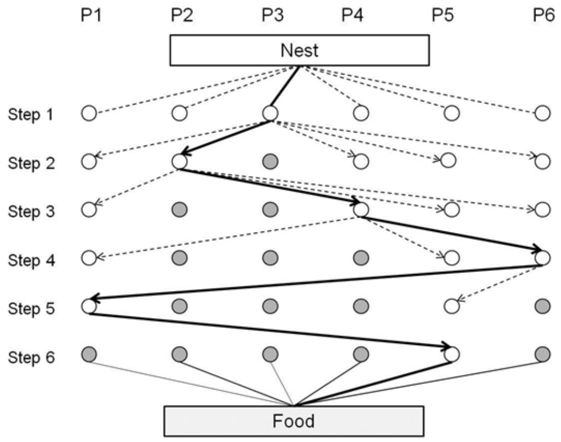

Figure 6: Modified ACO for the Production Planning Optimization

The bold arrows indicate the chosen path of an ant and every position already visited is depicted grey and cannot be chosen any more. In every step the remaining possible paths to the next position are shown as arrows with dashed lines.

This ant's "way" and resulting cost function value are then used as pheromone information to direct the moves of "following" ants by influencing their stochastic decision making process.

The ACO algorithm applied to solve the production planning task by providing an optimized product sequence works with the following steps:

1. Input parameters for ACO: cost function to be optimized (C_{max}), number of ants (n), number of iterations (i), number of products (p), initial pheromone matrix (P_i).
2. Initialize an ant ($a_{n,i}$) with random starting position (pos_p). This position represents the first product in the sequence.
3. Move $a_{n,i}$ to the next position according to a probability factor (pf_p) (where pf_p for an already "visited" $pos_p = 0$) and the pheromone trail in P_i.
4. Repeat 3. until all pos_p are "visited" and thus a possible solution / product sequence is created for each $a_{n,i}$.
5. Evaluate each ant's fitness (cost function value).
6. The sequence of $a_{n,i}$ with the best value (s_{best}) represents the best way so far and is used to lay pheromone upon in its representation in P_i. Thus $a_{n,i}$ in following iterations are more attracted to this way / sequence.
7. Evaporate pheromone according to an evaporation factor (ef) with $0 < ef < 1$ to update P_i into P_{i+1} following equation (1):

   $$P_{i+1} = P_i * ef \quad (1)$$

   The value of ef influences how randomly $a_{n,i}$ are searching for an optimal product sequence. A low value will lead to a more random creation of new product sequences, a high value will lead to rather preserving the best sequence found so far and smaller changes thereof.
8. Repeat 2. – 7. until an exit criterion (maximum i, time limit, result quality, etc.) is met.

2.2.3. Particle Swarm Optimization

There are many examples in nature where animals are forming swarms for higher individual safety or better chances to locate food sources. In such a swarm each individual member searches for a food source by itself while staying within a certain range to its neighbours and thus maintain the swarm structure. Once a swarm member finds food it will move towards it, attracting other swarm members to move in the same direction. The more individuals move to a food source and thus influencing the motion of their swarm neighbours, the whole swarm will move to a certain location over time depending on the ratio of moving individuals compared to the swarm size.

The "swarm" in PSO consists of particles, possible solutions (e. g. product sequences as in the presented work) of a given optimization problem, that represent the individual swarm members in nature. During the iterations of the algorithm the particles are "flying" through the search space and due to a frequent update and comparison of the swarm's best sequence so far and each particle's current value of the cost function, move over time to the optimal solution of the given optimization problem.

The PSO algorithm adapted for the bakery scheduling problem works as follows:

1. Input parameters for PSO: cost function to be optimized (C_{max}), number of particles (or swarm size), number of iterations (i).
2. Initialize a swarm of particles (x_i) with random positions (p_i) and velocities (v_i).
3. Each x_i represents a possible product sequence and the predefined initial v_i defines how each of these sequences is changed from the initial to the next iteration.
4. Evaluate each particle's fitness (cost function value).
5. Compare particle's fitness with its personal best value (*pbest*), update *pbest* if current fitness value is better and set *pbest* position of the particle to the current position.
6. Compare particle's fitness to swarm's best value (*gbest*), update *gbest* if current fitness value is better and set *gbest* parameters to the according particle's parameters.
7. Change particle's velocity and position according to equations (2) and (3) respectively:

   $$v_i = v_i + \mathrm{C1} * (pbest_i - x_i) + \mathrm{C2} * (gbest - x_i) \quad (2)$$

$x_i = x_i + v_i$ (3)

C1 and C2 are two positive constants in the original PSO and in this application equal "1" since the velocity is used to create a new product sequence.

8. Repeat 3. – 6. until an exit criterion (maximum *i*, time limit, result quality, etc.) is met.

3. RESULTS

MGA, ACO and PSO were used separately to perform 21 separate optimization runs each. Figures 7 and 8 show the results as C_{max} and the reduction of C_{max} compared to the initial value, respectively. The objective function value of the initial product sequence, which is the representation of the real production schedule used in the modelled bakery, for C_{max} was 1,380 min.

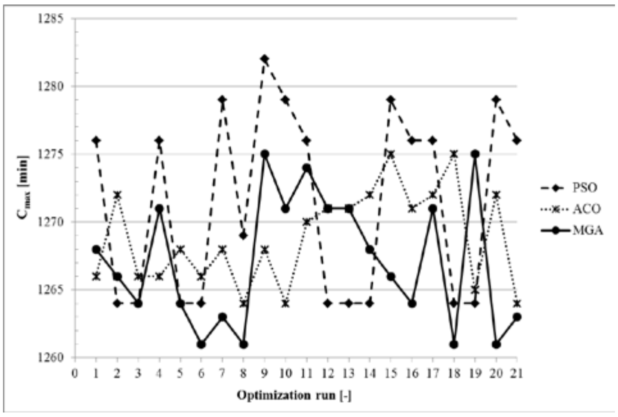

Figure 7: Optimization Results for C_{max} using Different Algorithms

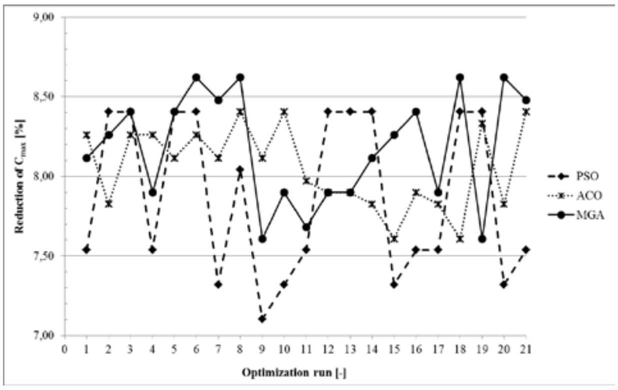

Figure 8: Reduction of C_{max} in % Compared to Initial C_{max} Value

The results show that each algorithm's optimization run found an optimized solution for C_{max}. With mean values for C_{max} of 1,267 ± 5 min (reduction of 8.18 ± 0.35 %) for MGA, 1,269 ± 4 min (reduction of 8.05 ± 0.26 %) for ACO and 1,271 ± 7 min, all three algorithms performed quite similar. To further analyse if significant differences in the algorithm performances can be obtained, a one-way ANOVA (**AN**alysis **O**f **VA**riance) with significance level $\alpha = 0.05$ was performed. A significant difference in the results is given if the resulting *p*-values of the ANOVA are smaller than the significance level α and the smaller the *p*-values the higher the significance of the differences.

The resulting *p*-values show that the results for optimizing C_{max} of MGA and ACO have no difference ($p = 0.18$), as well as the results of ACO and PSO ($p = 0.15$). The comparison of the results of MGA and PSO instead seems to show a difference with MGA performing better ($p = 0.03$).

The MGA provides the overall best solution of all optimization runs for C_{max} with a reduction of 8.62 % (or 119 min) in four out of 21 runs, PSO provides the overall worst solution with a reduction of 7.10 % (98 min). Still all algorithms even in their worst case performances provide significantly better results for C_{max} than the initial sequence. This means that the production goal of the analysed shift could be reached considerably faster, saving between 98 and 119 min of shift length depending on the optimization result, and thus saving a significant amount of man hours. This becomes especially clear if one has in mind that the reduction of shift length has to be multiplied with the number of involved staff members to calculate this reduction of man hours.

With a mean computational time of 749 ± 28 s, MGA tends to run slightly faster than ACO with a mean computational time of 775 ± 6 s and PSO with 783 ± 17 s. ANOVA results confirmed this, computational times of MGA and ACO ($p = 2.0\text{x}10^{-4}$) and MGA and PSO ($p = 2.7\text{x}10^{-5}$) show a highly significant difference, respectively. The differences in the computational times of ACO and PSO could not be determined clearly, the ANOVA result provided a *p*-value of $p = 0.0455$ and is thus just slightly below the significance level of $\alpha = 0.05$.

For a better evaluation of the obtained results, additional optimization runs without time restrictions where performed using 2,000 individuals and 500 generations in MGA and 2,000 ants/particles and 500 iterations in ACO/PSO. All three algorithms provided 1,261 min as best result for C_{max}.

4. DISCUSSION

The obtained results are distributed in a certain range, indicating that the sequences obtained in most of the optimization runs have to be local optima. The overall best result (1,261 min) obtained by using MGA seems to represent the global best solution for the problem, as the optimization runs without time restriction also provided this result for all employed optimization methods. This objective function value, that occurred seven times (in four MGA runs and the three optimization runs without time restriction), was provided by four different product sequences.

As with every modelling and optimization task, the quality of the obtained results is directly dependent on the quality and completeness of the data it is based upon. Therefore the optimization results presented have to be regarded critical due to the limited amount of real production data available. The optimization of C_{max} is based only on the data for the bread and bread roll production lines of the case study bakery, possible interactions with other production departments at the baking stages would probably affect the presented results, like the decrease of C_{max} by 7.10-8.62 % (or 98-119 min). Here more data has to be collected from the bakery involved in this work. Also the participation of additional companies providing their production data would further enhance the progress of this project.

Nevertheless even the worst case results for the optimization of C_{max} show significant benefits compared to the initial product sequence's objective function value, regardless of the applied methods.

Since the initial product sequence represents the real production schedule, the calculated results for the respective starting and end times of the products using the model presented in section 2.1.1. were used for verification. These results were checked and approved to be correct in respect to the real production progress by the shift leaders of the bakery modelled in this work. Thus the model used seems to be valid and provides reliable simulation results.

5. CONCLUSION AND OUTLOOK

The application of numerical modelling and optimization algorithms to develop a production planning procedure capable of solving the scheduling task in a bakery proved to be successful. A model of the production processes was designed in MATLAB, that schedules the workflow of a given product sequence according to defined decision parameters. An MGA, an ACO and a PSO have been used separately to solve the optimization task with respect to the optimal makespan of the production.

The applied methods proved capable of solving the given optimization problems in a computational time restricted to a maximum of 15 min, thus providing a scheduling tool that can be employed in a limited time frame prior to a production shift start.

After appropriate customization of the production model it is now possible for decision-makers in baking companies to approach their scheduling task in a fast and promising way capable for usage in praxis, based on the developed modelling and optimization procedure. Since even the worst case results of the applied methods yielded significant benefits compared to the results given by the initial product sequence, using such a mathematical procedure would most probably lead to a considerable increase of baking companies‘ efficiency.

Running the developed procedure on a PC with higher performance, to calculate more iterations or bigger ant/particle swarms or population sizes in the same computational time restriction, or simply allowing more time for the calculation, would be a promising way to further enhance the obtained results.

Another succeeding step will be to investigate the whole production of the modelled bakery, covering all existing different production departments to provide a companywide optimization approach.

Simultaneously we are also trying to convince additional bakeries to become new project partners and to provide their production information for new modelling and simulation studies.

ACKNOWLEDGEMENTS

This research has been funded by CSM Deutschland GmbH. Mr.

Hermann-Josef Michaelis (DialogMarketing GmbH) supported the collection and availability of the production data. The utilized production data were provided by a baking company in Frankfurt/Main, Germany. This research was not further influenced besides the financial and data collection support.

M. Sc. Marc Stanke, Dipl.-Chem. Olivier Paquet-Durand and M. Sc. Walid B. Hussein contributed with their support in the programming of the optimization algorithms.

REFERENCES

Chandra Mohan, B., Baskaran, R., 2012. A survey: Ant Colony Optimization based recent research and implementation on several engineering domain. *Expert Systems with Applications*, 39, 4618-4627.

Dorigo, M. Gambardella, L.M., 1997. Ant colonies for the travelling salesman problem. *BioSystems*, 43, 73-81.

Dorigo, M., 1992. *Optimization, learning and natural algorithms*. Thesis (PhD). Politecnico di Milano, Italy.

Dorigo, M., Di Caro, G., Gambardella, L.M., 1999. Ant Algorithms for Discrete Optimization. *Artificial Life*, 5, 137-172.

Eslami, M., Shareef, H., Khajehzadeh, M., Mohamed, A., 2012. A Survey of the State of the Art in Particle Swarm Optimization. *Research journal of Applied Sciences, Engineering and Technology*, 4 (9), 1181-1197.

Fereidoonian, F., Mirzazadeh, A., 2012. A genetic algorithm for the integrated scheduling model of a container-handling system in a maritime terminal. *In Proceedings of the Institution of Mechanical Engineers Part M: Journal of Engineering for the Maritime Environment*, 226 (1), 62-77.

Fondrevelle, J., Oulamara, A., Portmann, M.-C., Allahverdi, A., 2009. Permutation flow shop with exact time lags to minimise maximum lateness. *International Journal of Production Research*, 47 (23), 6759-6775.

Garey, M., Johnson, D., Sethy, R., 1976. The complexity of flow shop and job shop scheduling. *Mathematics of Operations Research*, 1 (2), 117-129.

Goldberg, D.E., Lingle, R., 1985. *Alleles, loci, and the traveling salesman problem. Proceedings of the 1st international conference on genetic algorithms and their applications*, pp. 154-159. July 24-26, Pittsburgh (Pennsylvania, USA).

Gómez-Gasquet, P., Andrés, C., Lario, F.-C., 2012. An agent-based genetic algorithm for hybrid flowshops with sequence dependent setup times to minimise makespan. *Expert Systems with Applications*, 39, 8095-8107.

Grabowski, J., Pempera, J., 2000. Sequencing jobs in some production system. *European Journal of Operational Research*, 125, 535-550.

Gupta, J.N.D., Stafford, E.F., 2006. Flowshop scheduling research after five decades. *European Journal of Operational Research*, 169, 699-711.

Hall, N.G., Sriskandarayah, C., 1996. A survey of machine scheduling problems with blocking and no-wait in process. *Operations Research*, 44, 510-525.

Hejazi, S.R., Saghafian, S., 2005. Flowshop-scheduling problems with makespan criterion: a review. *International Journal of Production Research*, 43 (14), 2895-2929.

Holland, J.H., 1975. *Adaption in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence*. Oxford: University of Michigan Press.

Jackson, J.R., 1955. Scheduling a production line to minimize maximum tardiness. *Research Report 43*, Management Science Research Project, University of California, Los Angeles.

Johnson, S.M., 1954. Optimal two- and three-stage production schedules with set up times included. *Naval Research Logistics Quarterly*, 1, 61-68.

Kennedy, J., Eberhart, R.C., 1995. Particle swarm optimization. *Conference Proceedings Volume 4 1995 IEEE International Conference on Neural Networks*, pp. 1942-1948. Nov. 27-Dec. 1, Perth (Australia).

Lenstra, J.K., Rinnooy Kan, A.H.G., 1978. Complexity of scheduling under precedence constraints. *Operations Research*, 26, 22-35.

Lian, Z., Gu, X., Jiao B., 2008. A novel particle swarm optimization algorithm for permutation flow-shop scheduling to minimize makespan. *Chaos, Solitons and Fractals*, 35, 851-861.

Linn, R., Zhang, W., 1999. Hybrid flow shop scheduling: a survey. *Computers & Industrial Engineering*, 37 (1-2), 57-61.

Liu, H., Gao, L., Pan, Q., 2011. A hybrid particle swarm optimization with estimation of distribution algorithm for solving permutation flowshop scheduling problem. *Expert Systems with Applications*, 38, 4348-4360.

Minella, G., Ruiz, R., Ciavotta, M., 2008. A review and evaluation of multiobjective algorithms for the flowshop scheduling problem. *INFORMS Journal on Computing*, 20 (3), 451-471.

Mullen, R.J., Monekosso, D., Barman, S., Remagnino, P., 2009. A review of ant algorithms. *Expert Systems with Applications*, 36, 9608-9617.

Pan, Q.-K., Tasgetiren, M.F., Liang, Y.-C., 2008. A discrete particle swarm optimization algorithm for the no-wait flowshop scheduling problem. *Computers & Operations Research*, 35, 2807-2839.

Phanden, R.K., Jain, A., Verma, R., 2012. A Genetic Algorithm-based approach for flexible job shop scheduling. *Applied Mechanics and Materials*, 110-116, 3930-3937.

Pinedo, M.L, 2008. *Scheduling - theory, algorithms, and systems*. 3rd ed. New York: Springer.

Poli, R., Kennedy, J., Blackwell, T., 2007: Particle swarm optimization: An overview. *Swarm Intelligence I*, 33-57.

Raaymakers, W., Hoogeveen, J., 2000. Scheduling multipurpose batch process industries with no-wait restrictions by simulated annealing. *European Journal of Operational Research*, 126, 131-151.

Rajendran, C., 1994. A no-wait flowshop scheduling heuristic to minimize makespan. *Journal of the Operational Research Society*, 45, 472-478.

Ruiz, R., Vázquez-Rodriguez, J.A., 2010. The hybrid flow shop scheduling problem. *European Journal of Operational Research*, 205, 1-18.

Smith, W.E., 1956. Various optimizers for single stage production. *Naval Research Logistics Quarterly*, 3, 59–66.

Tasgetiren, M.F., Liang, Y., Sevkli, M., Gencyilmaz, G., 2007. A particle swarm optimization algorithm for makespan and total flowtime minimization in the permutation flowshop sequencing problem. *European Journal of Operational Research*, 177, 1930-1947.

Tavares Neto, R.F., Godinho Filho, M., 2013. Literature review regarding Ant Colony Optimization applied to scheduling problems: Guidelines for implementation and directions for future research. *Engineering Applications of Artificial Intelligence*, 26, 150-161.

Ventura, J.A., Yoon, S.-H., 2013. A new genetic algorithm for lot-streaming flow shop scheduling with limited capacity buffers. *Journal of Intelligent Manufacturing*, 24 (6), 1185-1196.

Vignier, A., Billaut, J.C., Proust, C., 1999. Hybrid flowshop scheduling problems: state of the art. *Rairo-Recherche Operationelle-Operations Research*, 33 (2), 117-183.

Zentralverband des Deutschen Bäckerhandwerks e. V., 2012. Available from: http://www.baeckerhandwerk.de/baeckerhandwerk/zahlen-fakten [accessed 05.11.2013].

Zhang, Y., Zhou, Z., Liu, J., 2012. The production scheduling problem in a multi page invoice printing system. *Computers & Operation Research*, 37, 1814-1821.

Ziaeifar, A., Tavakkoli-Moghaddam, R., Pichka, K., 2012. Solving a new mathematical model for a hybrid flow shop scheduling problem with a processor assignment by a genetic algorithm. *International Journal of Advanced Manufacturing Technology*, 61 (1-4), 339-349.

AUTHORS BIOGRAPHY

Florian T. Hecker was born 1979, finished his apprenticeship as a baker in 2002 and afterwards studied Food Technology at the University of Hohenheim, Stuttgart, Germany, where he received his Engineer‘s Degree in 2008. In May 2008 he started working as a researcher and PhD student in the Department of Process Analytics and Cereal Technology, Institute of Food Science and Biotechnology, University of Hohenheim, Stuttgart, Germany. His research interests are in production optimization, scheduling theory and image processing based quality evaluation systems.

Bernd Hitzmann studied Physics at the University of Hannover, Germany, where he got also his PhD from the Faculty of Chemistry. Thereafter he worked as Post-doc at the California Institute of Technology, Pasadena, USA, as well as the Research Center of Asea Brown Boveri, Heidelberg, Germany. He got is habilitation in Technical Chemistry at the University of Hannover, Germany. Now he is head of the Department of Process Analytics and Cereal Technology, Institute of Food Science and Biotechnology, University of Hohenheim, Stuttgart, Germany. His research areas are: Design of experiments, spectroscopy for process monitoring, mathematical modelling, data evaluation, control, automation, and optimization of process in biotechnology and especially in cereal technology.