A Constraint Programming Model for Food Processing Industry: a Case for an Ice Cream Processing Facility

Journal:	*International Journal of Production Research*
Manuscript ID	TPRS-2017-IJPR-2425.R1
Manuscript Type:	Original Manuscript
Date Submitted by the Author:	31-May-2018

Complete List of Authors:	Wari, Ezra; Lamar University, Industrial Engineering Zhu, Weihang; Lamar University, Industrial Engineering

Keywords:	SCHEDULING, CONSTRAINT PROGRAMMING
Keywords (user):	Ice Cream Processing, Food Processing Scheduling

Dear Editor-in-Chief, Editor and Reviewers:

The reviewer comments are very helpful for improving the papers. We have carefully followed the reviewer comments during the revision process. The major revision paragraphs are highlighted in red color in the paper. Please find below the response to the reviewer comments.

We greatly appreciate your consideration and hope this revision can meet your expectation.

Sincerely

Authors

Reviewer: 1

Comments to the Author
The article tackles an interesting realistic scheduling problem in the food processing industry.

After a literature review in the domain, it describes the scheduling problem and proposes and compares two formulations: a MIP (described in a former paper) and a CP one. I think this work is very relevant for the Journal of Production Research.

This being said, there are two main issues in the manuscript that I think should be fixed before the article could be accepted.

1/ Problem and CP model description

First, the description of the problem and of the CP formulation should be improved. It is difficult to have a global vision of the problem, in spite of the description of the facility on Figure 1. From a model formulation perspective, it would be useful to add a figure that describes the different operations that are modeled with the interval variables of the CP model and the overall process, with the precedence constraints between operations. That would be complementary with Figure 1.

Answer: Thank you for the comments. The model is indeed complex. We have added a figure (Figure 2) and a descriptive paragraph (page 8, paragraph 2) to help readers get an overview of the whole model.

Another problem is that the article assumes that the reader is familiar with the modeling concepts used in CP Optimizer (optional interval variables, sequence variables with transition matrices,...). I do not think you can make this assumption and it would be necessary to add a small section (but more detailed than the current one on p. 3) that describes these modeling concepts and refers to some papers (like [1] and [2]) for more details.

[1] P. Laborie, J. Rogerie. Reasoning with Conditional Time-intervals. Proc. 21th International FLAIRS Conference (FLAIRS 2008)
[2] P. Laborie, J. Rogerie, P. Shaw, P. Vilim . Reasoning with Conditional Time-intervals, Part II: an

Algebraical Model for Resources. Proc. 22th International FLAIRS Conference (FLAIRS 2009)

Answer: Thank you for the comments. We have added a general description on CP on page 1, and description of CP modeling using IBM ILOG on page 9 paragraph 1, paragraph 4, page 10 paragraphs 1 – 3, on page 12 paragraph 1, and Figures 2 – 5. Both papers have been cited in the paper.

Some more detailed comments on the CP model:

* I would change the title of section 3.2 ("CP Solution Model") into something like "CP Formulation" or "CP Model". The "Solution" term seems to imply that your CP approach is more than just a declarative model but that you also have to write some resolution algorithm. Which is of course not the case as here you are using a purely declarative CP formulation of the problem, just like a MIP, and you rely on the automatic search of CP Optimizer.

Answer: Thank you for the comments. We have made the correction accordingly.

* In the model, in constraints (7), as WaitingProcessV_ib is an optional interval, when you use constraint startAtEnd(WaitingProcessV_ib,AgeProcessV_ib), it means that if interval WaitingProcessV_ib is absent, the temporal constraint will not be enforced, and in particular if you have a chain of precedence constraints going through this startAtEnd, this chain will be "broken" in case WaitingProcessV_ib is absent. Is it really what you want to model? For a better understanding of the global problem, this is typically where a Figure showing all the operation/intervals and the temporal constraints would really help.

Answer: Thank you for the comments. We have changed WaitingProcessV_ib to be compulsory.

* I'm not sure to understand constraint (19). The constraints basically says (C is a condition): (C && endOf(AgeProcessV_ib)>n*Week) => endOf(AgeProcessV_ib)==n*Week. As this implication is never feasible, it just seems to mean that the left hand side of the implication should always be false. So !C || endOf(AgeProcessV_ib)<=n*Week. I don't see which constraint you want to formulate here. Or may be it is a forbidEnd constraint that prevents the interval to end during a week-end?

Answer: Thank you for the comments. We have replaced constraint 19 with three new constraints (19, 20 and 21). We defined the size of the aging interval ($AgeProcessV_{ib}$) to be the normal size when it starts and finishes on the week day (19) and to include all hours up to the start of the subsequent week when the aging interval is between two consecutive weeks (20).

* Constraint (20) could probably be formulated using an endBeforeStart constraint. This can be more efficient.

<u>Answer</u>: Thank you for the comments. We have changed the model to create a processing chain. The constraint is no longer needed, hence it has been deleted

* Constraint (22) could probably be formulated as lengthOf(VesselAssign_ib)<72. This can be more efficient.

<u>Answer</u>: Thank you for the comments. We have made the correction accordingly.

2/ Experiments

The main problem with the comparison of the MIP and CP model in section 4.1 is that it is not clear if the 2 models model the same problem. In the beginning of the section, you say that the MIP model had to be slightly adapted to be comparable to the CP one (as CP handles only integer). So one can conclude that after these adaptations, the two formulations represent the same problem. But on Table 1, as the reader understands that any time lower than 600s means that the approach proves optimality, how can it be that, for instance for the smallest instance 1, the MILP and CP model do not find the same optimal makespan (129 v.s. 136) ? And the following sentence seems to suggest that the formulations do not represent exactly the same problem: "Overall, the difference ranged from +14% to -15% which can be attributed to model variation". This should be clarified. If the formulations are for two different problems, then how can you compare them? Furthermore, on Table 1, you should clearly identify the instances that were solved to optimality by each approach.

<u>Answer</u>: Thank you for the comments. We used the same problem to test the models. The MILP attained better makespan values than the CP. By including the fail limit, any result attained **can not** be guaranteed to be optimal. We have removed fail limits in this revision and optimization runs did not complete with in the 600s, hence the CP model results **can not** be guaranteed to be optimal. We have tried to clarify this in the paper.

Also, it is not clear why the CP models are using a fail limit *and* a time limit. Why not just using a time limit? The fail limit does not really make sense from an applicative point of view.

<u>Answer</u>: Thank you for the comments. We have removed the fail limit.

The tentative of explanation of why the CP model performs better than the MILP in section 4.1 is not really convincing. The fact the MIP has a much larger number of constraints and variables is not enough as MIP and CP technologies work very differently for solving the problem. So, at minimum, you should be more careful for giving this as an explanation.

<u>Answer</u>: Thank you for the comments. We have made the correction. CP is better for large problem sizes and MILP model (with all its limitation) is the better model for small size problems.

3/ Additional comments:

* In the abstract: "CP is a mathematical optimization tool for solving scheduling problems". This is too restrictive: beside scheduling, CP can be used to solve many other combinatorial problems!

Answer: Thank you for the comments. We have made the correction.

* Nomenclature: I think it is a good idea to have some appendix like this. But right at the beginning of the article is not the right place I think. I would add it as an appendix in the end. Furthermore, the description of the "weekend break step function" that is using the (non-described) OPL syntax is not really understandable for a non-expert reader. I would give a more mathematical description of this step function.

Answer: Thank you for the comments. We have made the correction.

* In general, the typesetting of the mathematical equations is not very nice in the paper. I would suggest to use LaTeX.

Answer: Thank you for the comments. We have used MathType to write the equations.

* On page 3: "ILOG IBM CPLEX" -> "IBM ILOG CP Optimizer". I would insist on CP Optimizer as you are solving the CP model with this engine, not with CPLEX (MIP) engine.

Answer: Thank you for the comments. We have made the correction.

* In the literature review, I would not classify the neural net approaches as mathematic methods. First, in the cited reference, the article does not solve the problem (unlike the Branch&Bound, MILP, CP or heuristic methods): NN is only used to "estimate" the objective value. And I would not qualify this method as "mathematical" in this context. So I would rather remove it from the literature review.

Answer: Thank you for the comments. We have removed it.

* In the conclusion, the fact that the CP formulation is limited to integers is not a big problem I think as you can scale the problem as you want and the performance of the resolution does not really depend on the scaled factor.

Answer: Thank you for the comments. We have remove the explanation.

Reviewer 2

Review Report for the Manuscript Entitled " A Constraint Programming Model for Food Processing Industry: A Case for an Ice Cream Processing Facility"

This paper proposes a Constraint Programming (CP) model for scheduling processing operations for an ice cream facility that addresses a combined discrete and continuous production system. Pasteurization, dehydration, and freezing are a few examples of continuous processes whereas packing is an example of discrete processes.

The CP model is compared with a MILP present in the literature. The computational results indicate that CP performs better that the MILP for large size problems.

I suggest a major revision for this manuscript due to the following issues:

1) The literature review has been done by classifying the past literature about the scheduling problems in food processing industries as those that consider theoretical-based problems on one side and as those that consider industrial case problems on the other. This type of classification seems awkward. A table that summarizes the past literature with the problem specific characteristics could be easier to observe what has been done up to now. The characteristics of the studied problem different from the ones studied in related literature should be emphasized. There is no information about the features of the problem that is being studied and how this problem encompasses the features of scheduling problems in food industries as an overall.

Answer: Thank you for the comments. We have removed the classification, added a summary table and emphasize the features of the proposed model.

The relevant should be expanded.

Answer: Thank you for the comments. We have added the contribution of the paper from literature review perspective on page 5, paragraph 1.

There are missing references such as follows:

- Efficient mathematical frameworks for detailed production scheduling in food processing industries, Computers & Chemical Engineering, Volume 42, 11 July 2012, Pages 206-216, Georgios M. Kopanos, Luis Puigjaner, Michael C. Georgiadis
- Multi-week MILP scheduling for an ice cream processing facility, Computers & Chemical Engineering, Volume 94, 2 November 2016, Pages 141-156
- Real-world production scheduling for the food industry: An integrated approach, Engineering Applications of Artificial Intelligence, Volume 25, Issue 2, March 2012, Pages 222-228, Tony Wauters, Katja Verbeeck, Paul Verstraete, Greet
- Planning and scheduling of the make-and-pack dairy production under lifetime uncertainty, Applied Mathematical Modelling, Volume 51, November 2017, Pages 129-144 Çağrı Sel, Bilge Bilgen, Jacqueline Bloemhof-Ruwaard

Answer: Thank you for the comments. We have added references.

2) The authors indicate that their proposed CP model had obtained comparable results for small size problems to those obtained from the mixed integer linear programming model published by Wari and Zhu (2016). For large size problems the CP model has managed to find solutions for all problem instances whereas the MILP model could not find any solution within the time limit.

   In Section 3, it is said that the paper takes on the challenge of considering the facility at its full size as described in Bongers and Bakker (2006). However, it compares the proposed CP model to the MILP model developed by Wari and Zhu (2016). It is not clear which model is really being studied.

   <u>Answer</u>: Thank you for the comments. Our primary goal for this study is to solve the scheduling problem for the full capacity of the facility and we achieved this goal in this paper with CP. However, readers may wonder how good the CP model is in terms of optimization result quality and speed. We use the same problem for both models to compare and show these qualities of the proposed model.

3) It is also required that the problem which is being studied should be formulated by the MILP approach as well.

   While the CP model is being developed, its constraints should be explained given their correspondence to the counterpart constraints in the MILP model. This one-to-one mapping is especially important in the understanding of the proposed model.

   <u>Answer</u>: Thank you for the comments. We have added a summary of the MILP model used to compare the proposed model. However, we could not create a one-to-one correlation between the two models as the two implemented different approaches. We focused on the quality of the result and speed of optimization of each model for the same data.

4) The validation of the CP model should be done by solving a small size problem using the MILP model. It can be seen from the computational results on page 17 in Table 1 that the model could not give the same makespan values for the same problem instances. For example, for data set 1 for the problem instance 1 the CP model finds 136 hrs as a makespan value, whereas the MILP model finds this value 129 hrs. When runtime 64 s is given for solving the CP model, it is assumed that the CP model has solved the problem optimally before the time limit 600 s. Therefore, the optimal solution values of the models are not the same, which imply that the CP model could not properly reflect the constraints of the MILP model.

   <u>Answer</u>: Thank you for the comments. We have made the correction. CP is better for large problem sizes and MILP model (with all its limitation) is the better model for small size problems. With the different optimization run, we tried to show how close to optimal the CP model can achieve .

5) The production process should be described in more detail. The type of the scheduling problem handled should be indicated explicitly as the flexible flow shop scheduling problem. The processing restrictions should be given, and the continuous and discontinuous characteristics of the production process should be pointed out, and the differences from the discrete flexible flow shop scheduling problem should be given.

Answer: Thank you for the comments. We have added more details about the production processing distinctly showing the continuous and discrete production stages.

6) There are also some English syntax errors. For example, "makespan " is written as "make-span" in all the places.

On page 6, lines 12 and 13,

"...a wide range of subsector" should be replaced with " "...a wide range of subsectors".

"A summary of optimization approach for both ...formulation" should be replaced with "A summary of optimization approaches for both ...formulations".

On page 6, lines 49 through 51, "... In this algorithms..." should be replaced with "...In this algorithm..."

The English language should be revised.

Answer: Thank you for the comments. We have made the correction and revised paper for the syntax. We have added fixed many other typos in this revision.

AE comments:

The manuscript considers an interesting scheduling problem in the food processing industry. Constraint Programming (CP) approach is proposed and its result is compared with MILP result. It claims that CP outperforms the MILP for large size problems.

Two reviewers recommend major revision with very detailed comments, and thus the authors should carefully revise the paper according to reviewers' comments. The following is the overview of the comments by reviewers.

1. The problem description should be carefully written.
The clear description of the problem will help readers understand the model. If there are some different features from existing literature dealing with food processing, it may be helpful for readers to understand better.
Answer: Thank you for the comments. We have added more description including figures to help readers understand better the production process and problem model.

2. CP formulation should be clearly written.
Some readers may not be familiar with CP approach. Thus, the authors should mention the concept of CP step by step. Moreover, since the authors use the commercial CP solver, the contribution of this manuscript is focused on the CP formulation. Thus, the authors should explain the proposed CP formulation carefully.
Answer: Thank you for the comments. We have discussed the CP optimization approach, in general (what distinguishes it from other methods) and the commercial solver IBM ILOG CP, in particular. We have also discussed the built-in functions and constraints in the commercial software.

3. The results between CP and MILP should be compared in a fair manner.
As two reviewers pointed out, it is not clear whether the same problems are compared and whether the same results are compared. It is critical to evaluate the performance of CP.
Answer: Thank you for the comments. We have revised the results of the experimental runs and conclusions we derived from these results. Our comparison is only for the quality of results and speed of the two models.

A Constraint Programming Model for Food Processing Industry: A Case for an Ice Cream Processing Facility

Ezra Wari, Weihang Zhu*
Department of Industrial Engineering
Lamar University
Beaumont, TX 77710, USA
*Corresponding author email address: Weihang.Zhu@lamar.edu

Abstract

This paper presents a Constraint Programming (CP) scheduling model for an ice cream processing facility. CP is a mathematical optimization tool for solving problems either for optimality (for small-size problems) or feasibility (for large-size problems). For practical scheduling problems, a single CP solution model can be used to optimize daily production or production horizon extending for months. The proposed model minimizes a makespan objective and consists of various processing interval and sequence variables and a number of production constraints for a case from a food processing industry. Its performance was compared against a Mixed Integer Linear Programming (MILP) model from the literature for optimality, speed, and competence using the partial capacity of the production facility of the case study. Furthermore, the model was tested using different product demand sizes for the full capacity of the facility. The results demonstrate both the effectiveness, flexibility, and speed of the CP models, especially for large-scale models. As an alternative to MILP, CP models can provide a reasonable balance between optimality and computation speed.

Keywords: Constraint Programming; Scheduling; Ice Cream Processing; Food Processing Scheduling

1 Introduction

Companies put a significant effort to utilize production resources efficiently and effectively in their manufacturing activities. Scheduling methods improve resource utilization by providing platforms for planning, managing, and controlling resources. This paper presents a scheduling tool using Constraint Programming (CP) for a scheduling problem from the food processing industries.

Scheduling problems in the food processing industries must address a combined discrete and continuous production system. Pasteurization, dehydration, and freezing are a few examples of continuous processes whereas packaging process is a good example of discrete processes in the industry. In addition to the optimization objectives (such as production cost/profit, makespan, earliness or lateness) and constraints (assignment of tasks to machines, sequencing and/or timing of tasks, other facility related constraints) they have in common with discrete production systems, continuous systems require decisions on selection and size of processing batches (Harjunkoski, et al., 2014). The perishable nature of food products would be the other factor to consider in scheduling problems in the industry. This factor constrains the manufacturing process to be completed within a limited time window frame. Hence optimization models not only need to define a constraint for this time window but also should complete the optimization run under this time limit. Time for decision making, implementation, and any corrective action would have to be included in this time window as well.

Several mathematical programming techniques have been proposed to solve optimization problems (including scheduling). Some of these techniques include Linear Programming, Mixed Integer Linear Programming (MILP), Dynamic Programming (DP), and CP (Rossi, Van Beek, & Walsh, 2006; Apt, 2003). Among these programming methods, CP is the only technique which integrates a computer program in its solution model formulation. CP was first developed as a tool for solving combinatorial problems in the artificial intelligence and computer science field of study. Problems in this method are formulated by defining the constraints on the decision variables and then solving these problems using computer programs (procedures) (Bockmayr & Hooker, 2005; I. & JF., 2013). Other mathematical programming approaches only incorporate the declarative part of the formulation where the constraints and the objective of a problem are simply defined. By considering the solving procedures, CP can guide the solution search based on problem setups to improve its effectiveness.

CP utilizes an approach where a solution space is reduced using constraints/restrictors before solving the problem with different mathematical programming methods (Rossi, Van Beek, & Walsh, 2006; Apt, 2003). Problems can be formulated as Constraints Satisfaction Problem where CP aims at finding a feasible solution to the problem, or as Optimization Problem where it tries to find the optimal solution based on a given objective. Scheduling problems constitute one application area for the implementation of CP. Restriction due to manufacturing environments and resource availabilities can be formulated as constraints, and the production scheduling problem can be solved as either a constraint satisfaction problem or an optimization problem.

The proposed CP model, in this paper, minimizes a makespan objective for a medium size food processing facility (ice cream processing). In our CP model, interval and sequencing variables are created to define the start and completion of processing times and order of processing in a specific machine. Constraints for these variables included no-overlap of processing time, selection and assignment of a task to machines, interlink and order of processing stages for all product types, product processing order in the packaging lines, and cease of production over the weekends. The first experimental run is used to compare the model against an MILP solution model published by Wari and Zhu (2016) for a reduced size of the case study. The model attained makespan values comparable to those from the MILP model for small size problem instances. For more complex problems, it performed better by solving instances which the MILP model was unable to solve. In the second experimental run, the model solves two demand problem sets for the full-size production facility. It is worth noting that before this paper, no other researchers have been able to solve this problem at its full scale. Two batch sizes were used to break total products demands and a schedule for month's production horizon is reported.

The rest of the paper is organized as follows. The literature review on scheduling approaches for food processing industries and CP method is presented immediately after this introduction section. The proposed model is described in the third section. Section four discusses the experimental run results of the model. Finally, concluding remarks are given in the last section.

2 Literature Review

Scheduling problems in the food processing industries can be formulated by adopting either a general machine or production system layout, or a specific industrial case study problem approach. The most common general system layout problem formulation is given by a no-wait Flow Shop (FS) and Flexible Flow Shop (FFS) manufacturing setup. Industrial application cases covered a wide range of subsectors. This section presents a brief literature review of these approaches.

Various heuristic and metaheuristic approaches have been utilized to optimize scheduling problems with general system layout. Wang & Liu proposed a Genetic Algorithm (GA) model where jobs were coded as genes in the solution model (Wang & Liu, 2013). In this algorithm, jobs were defined as genes and schedule as chromosomes for a two-stage production process. To utilize the best features of multiple metaheuristic approaches, a number of publications adopted hybrids of heuristics/metaheuristics methods. Jolai, Rabiee, & Asefi (2012) proposed a hybrid Simulating Annealing (SA - Population-based SA) and Imperialist Competitive Algorithm (ICA) approach in which the earlier explored the solution space whereas the later exploited the neighborhoods. Other similar mixed approaches include Moradinasab, Shafaei, Rabiee, & Ramezani (2013) who developed models using ICA, Ant Colony Optimization (ACO), and GA; Zhou and Gu (2009) who integrated GA and Gaming Theory; and Samarghandi and ElMekkawy (2012) who presented a hybrid Tabu Search (TS) – Particle Swarm Optimization (PSO) approach. Ye, Li, & Miao (2017;

2016) proposed two heuristics methods (based on average idle and departure time) for a no-wait FS. Nagano, Miyata, & Araújo presented a constructive heuristic where the scheduling problem is broken-down into smaller sized problems before being optimized (Nagano, Miyata, & Araújo, 2015). Overall heuristic and metaheuristic dominate the approaches in the literature since most of no-wait FS/FFS scheduling problems were formulated as NP-Hard. However, few mathematical methods can be found which would include a branch and bound presented by Wang, Liu, & Chu (2015) for an FFS manufacturing setup.

Scheduling problems specific to the food processing facilities adopted similar metaheuristics, mathematical, or hybrid methods. The application of GA could be for whole production setup as presented by Shaw et al. (2000), and Heinonen & Pettersson (2003), or specific processing stage (the filling line of dairy plant) as proposed by Gellert, Höhn, & Möhring (2011) or production function (cost of distribution) for the case of Karray, Benrejeb & Borne (2011). In all cases, production constraints such as processing stages and precedence, clean-up/sanitization, machine capacity, and processing time were formulated into the problems. Banerjee et al. (2008) presented an Artificial Bee Colony (ABC) metaheuristic model for solving a multi-objective scheduling (optimal cost and risk levels) problem for a milk processing industry. Combined metaheuristics approach includes the two publications by Hecker et al. (2013; 2014) which adopted GA, ACO, PSO, and Random Search algorithms. Linear Programming, particularly MILP, dominate the mathematical approaches for specific food processing scheduling solution models. Bongers and Bakker (2006), Kopanos, Puigjaner and Georgiadis (2011), Kopanos, Puigjaner and Georgiadis (2012), and Wari and Zhu (2016) proposed MILP models for a simplified ice cream processing scheduling problem (the case study for this paper as well) with a makespan optimization objective. Bongers and Bakker (2006), Kopanos, Puigjaner and Georgiadis (2011), Kopanos, Puigjaner and Georgiadis (2012) integrated heuristics to supplement the limitation of the MILP approach such as long optimization runtime, shorter production scheduling horizon and few numbers of product types in the scheduling problem. Wari and Zhu (2016) presented an MILP model for multi-week production horizon with proper weekend break-up points and clean-up sessions. These four publications considered part of the production process of the facility and a smaller number of products. The proposed CP model in this paper presents optimization model for a larger number of products processed using the full capacity of the facility. It also presents a new mathematical approach for solving scheduling problems with the combined continuous and discrete production system. Other MILP models publications include Doganis and Sarimveis (2007; 2008a; 2008b) and Kopanos, Puigjaner, and Georgiadis (2009) who developed models to optimize production cost (yogurt processing facility), Sadi-Nezhad & Darian (2010) who presented a model to optimize production capacity (juice processing facility), and Liu, Pinto, and Papageorgiou (2010) who proposed a model to maximize profit (an edible oil manufacturing facility). New approaches, such as chance-constraint programming and a combined local search and machine learning methods, have been used to optimize industrial scheduling problems. For example, Wauters et. al. presented a scheduling model where data for different food processing features were ultilized by search heuistics to attain better result (Wauters, Verbeeck, Verstraete, Berghe, & De Causmaecker, 2012). Sel, Bilgen, & Bloemhof-Ruwaard demonstrated the application of chance-programming for a scheduling problem in a dairy processing facility (Sel, Bilgen, & Bloemhof-Ruwaard, 2017). The model applied chance-programming to quality decaying properties of dairy products.

Since no publication could be identified for specific applications in the food processing industries, the literature review for CP application has been expanded to include other manufacturing industries sectors. For machine scheduling application of CP, Novas and Henning (2012), Öztürk et al. (2012), and Zeballos, Quiroga, & Henning (2010) presented a makespan minimizing model for an automated wet-etch station (semiconductor manufacturing), flexible mixed-model assembly line, and machine-tool allocation and routing of products respectively. The constraints in all models include start/completion processing times, the processing order (for products) and resource availability. In shift scheduling problem (for employees), the optimization objective includes meeting demand requirements of labor and reducing costs such as overtime and under-utilized labor. Model constraints may be comprised of factors such as labor cost, workload balance, and allowable working hours. Publications for this instance included Han & Li (2014) – optimizing drivers and operators for a mass rapid transit train system, and Topaloglu & Ozkarahan (2011) – optimizing the schedule of medical residencies. CP can also be used for fleet scheduling optimization either in a distribution/logistics problem or an internal material handling system to minimize cost. Unsal & Oguz (2013) and El Hachemi, Gendreau, & Rousseau (2011) presented CP models to demonstrate this application. Routing, inventory, and combined planning and scheduling problem are a few other application areas for CP (Goel, Slusky, van Hoeve, Furman, & Shao, 2015; Zhang & Wong, 2012). Table 1 summarizes the key relevant publication in the food processing industry.

Table 1 Summary of relevant scheduling problem publication in the food processing industry

Publication	Formulation	Objective	Mathematical MILP	CP	B&B	Metaheuristic/Heuristic GA	SA	ICA	ACO	TS	PSO	ABC	Heuristic	Other
Wang & Liu, 2013	2-stage no-wait FFS	Makespan				x								
Jolai, Rabiee, & Asefi (2012)	No-wait FFS	Makespan					x	x						
Moradinasab, Shafaei, Rabiee, & Ramezani (2013)	No-wait FFS	Total completion time				x		x	x					
Zhou and Gu (2009)	No-wait FFS	Customer satisfaction				x								x
Samarghandi and ElMekkawy (2012)	No-wait FS	Makespan								x	x			
Ye, Li, & Miao (2017; 2016)	No-wait FS	Makespan											x	
Nagano, Miyata, & Araújo, 2015	No-wait FS	Total flow time											x	
Wang, Liu, & Chu (2015)	2-stage no-wait FFS	Makespan			x									
Shaw et al. (2000)	Batch/Continuous (no-wait FS)	Various production costs				x								
Heinonen & Pettersson (2003)	Batch/Continuous (no-wait FS)	Various production costs				x								
Gellert, Höhn, & Möhring (2011)	Dairy processing facility	Total production cost				x								
Karray, Benrejeb, & Borne (2011)	Agro-food industries	Various production costs				x								
Banerjee et al. (2008)	Milk processing facility	Cost and risk levels										x		
Hecker et al. (2013; 2014)	No-wait FFS (Bakery facility)	Total production cost				x			x		x		x	
Bongers and Bakker (2006)	Ice cream processing facility	Makespan	x										x	
Kopanos, Puigjaner and Georgiadis (2011; 2012)	Ice cream processing facility	Makespan	x										x	
Wari and Zhu (2016)	Ice cream processing facility	Makespan	x											
Doganis and Sarimveis (2007; 2008a; 2008b)	Yogurt processing facility	Production Cost	x											
Kopanos, Puigjaner, and Georgiadis (2009)	Yogurt processing facility	Production Cost	x											
Sadi-Nezhad & Darian (2010)	Juice processing facility	Production Capacity												
Liu, Pinto, and Papageorgiou (2010)	Edible oil manufacturing facility	Profit	x											
Wauters et. al.	Packaging line	Makespan												x
Sel, Bilgen, & Bloemhof-Ruwaard (2017)	Dairy processing facility	Makespan												x

3 Problem Description and Solution Models

The case study for this paper was first proposed by Bongers & Bakker (2006) to improve production scheduling for an ice cream processing facility. Bonger & Bakker reduced the size of the production in the facility to three-stage processing with a fewer number of machines, and product mixes mainly due to the limitation of the optimization software. Later, relevant publications have expanded the number of product mixes and demands sizes, while keeping the problem at the reduced size. This paper takes on the challenge of considering the facility at its full size as described in Bongers & Bakker (2006).

3.1 Problem Description

As illustrated in Figure 1, production in the ice cream processing facility starts by mixing different ingredients of the ice cream based on receipts (Bongers & Bakker, 2006). This stage is assumed to have no resource limitation and hence excluded from the scheduling problem. Ice cream mixes are pasteurized first using two continuous-pasteurization units. These units expose the mixtures to a high temperature for a short period as the mixtures flow through the equipment. The products flow to the aging vessels where the quality of the ice cream mix is improved, and the mix is cooled-down. Aging vessels are refrigerated tankers with agitators and store ice cream mixtures for a receipt-based period. Each product mixes can be aged in a specific number of vessels. The ice cream mixes are cooled further using product-specific freezers for a predefined period. The production process concludes with the packaging of product mixes into different sizes and shapes at a rate specific to each product mix.

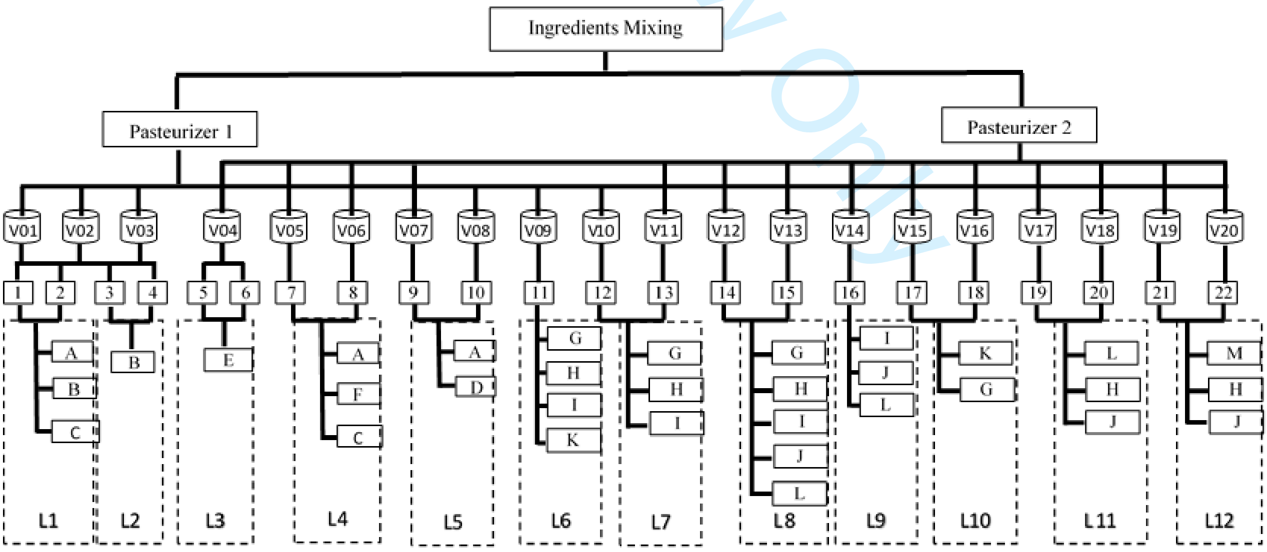


Figure 1. A medium size ice cream processing facility: V01 to V20 are aging vessels; Items 1 to 22 are freezers; Items L1 to L12 are packaging lines; Items A to M are product mix types (Source: Bongers and Bakker (2006)**)**

Product processing speeds have been defined in two methods in the problem. In the first method, machine's processing speed determines the processing rate. For example, the two pasteurization units process all product mixes at the rate of either 4500Kg/hr or 6000Kg/hr. The second method defines the processing speed based on the product mix type. Aging and packaging rate are good examples where product's intrinsic nature determines the processing rate. Most machines can process only a group of product mixes except for Pasteurizer 1 which has the capacity to process all products.

The processing flow of the ice cream product mixes encompasses several stages (Figure 2). In the first stage, pasteurization of the product mixes takes place at a rate based on the specific equipment under consideration. Processing inside the vessels consists of multiple steps. First, the vessels would have to be filled up before the aging commences. As a continuous process, the pasteurization and filling processes can be considered as overlapping processes (interval variables for the two would have the same value). Then product mixes are aged and either transferred to the freezer or stored depending on the availability of a freezer and packaging line. When there is no idle freezer or packaging line available, the product is temporarily stored in the vessel (creates waiting interval). When a freezer and packaging line is available to take the mix, continuous freezing and packaging processes commence. Similar to the filling step, the emptying step of a vessel is a continuous process that overlaps with the freezing and packaging stages. Freezing and packaging are independent and consecutive processes.

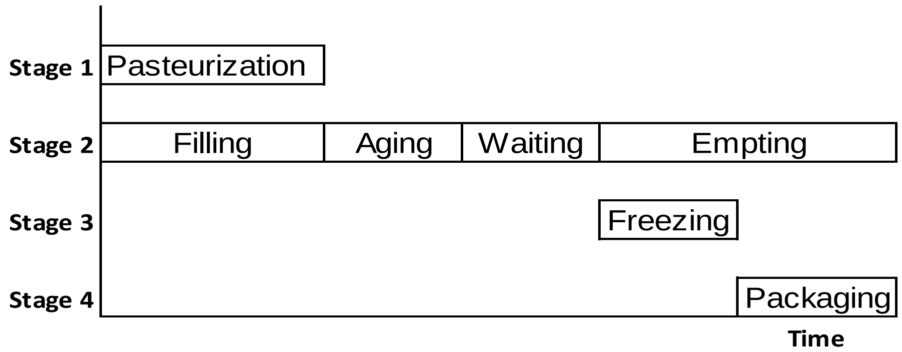


Figure 2. Processing time intervals for different stages of the ice cream production

3.2 CP model

Scheduling problems can be formulated using different approaches in CP. The most common approach is to define processes using bounding, binary, or both types of decision variables and then adopt constraint functions and propagating algorithms to solve the problem (Laborie & Rogerie, 2008). IBM ILOG CP optimization tool provides a different approach where the processes and respective sequences are formulated as decision variables. This tool incorporates different model formulation approaches which include global constraints construction, modeling layers creation (for processing activities and resources), and methods for generating optional processing activities (Laborie, Rogerie, Shaw, & Vilím, 2009). The proposed CP model creates processing time interval decision variables for all products in all stages. Sequencing decision variable defines the processing orders of products scheduled in each equipment. The complete model is described in this section.

Sets and Subsets

Sets create collections for product mix types, and machines for pasteurization, aging, and packaging whereas subsets memorize the assignments of processing machines for product mixes.

Parameters

Parameters define model constants such as processing speeds (filling, aging, emptying), changeover times, working week time horizons, and scheduled weeks whereas parameter functions formulate mathematical relationship among parameters.

Decision Variables

Two types of decision variables, intervals, and sequences are created to formulate different constraint relation which emulates the processing restrictions of the ice cream production. Interval decision variables consist of two types of variables. Process Intervals construct the processing time interval for each product mix in all assigned machines (*VesselProcess, FillProcess, FreezeProcess, EmptyProcess*). Assignment intervals select a process interval among multiples of alternatives to create a schedule for each product mix (*VesselAssign*, *FillAssign*, *FreezeAssign*, *EmptyAssign*).

Figure 3 shows the temporal and logical constraints of the interval variables. The chain processing step is shown with the series of time intervals (linked by straight arrows). *VesselAssign* variables span over all the intervals and align with *FillAssign* at the start and *EmptyAssign* at the end (linked by double-headed arrows). Filling and emptying steps of the vessel are equivalent to the pasteurization process and the sum of freezing and packaging processes, respectively.

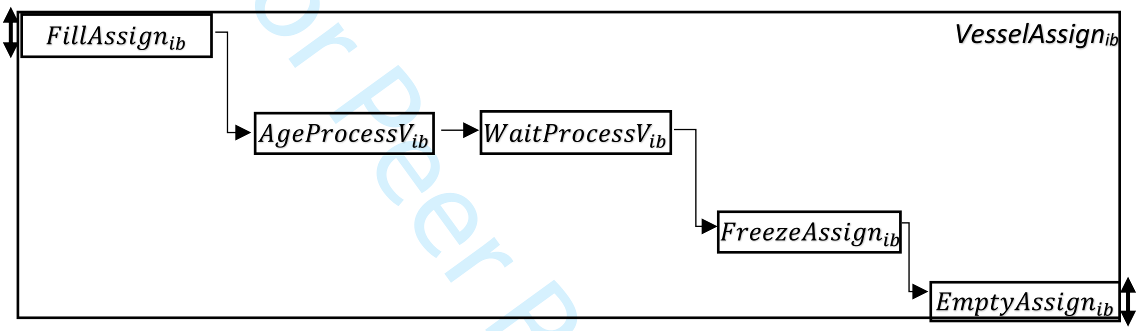


Figure 3. Temporal and spanning decision variables

The processes at all four stages of production can be performed by multiple equipment capable of processing a given product. Alternative constraints formulate these options where the process interval decision variables represent individual feasible options while the assign interval variables represent the selected options (see Figure 4).

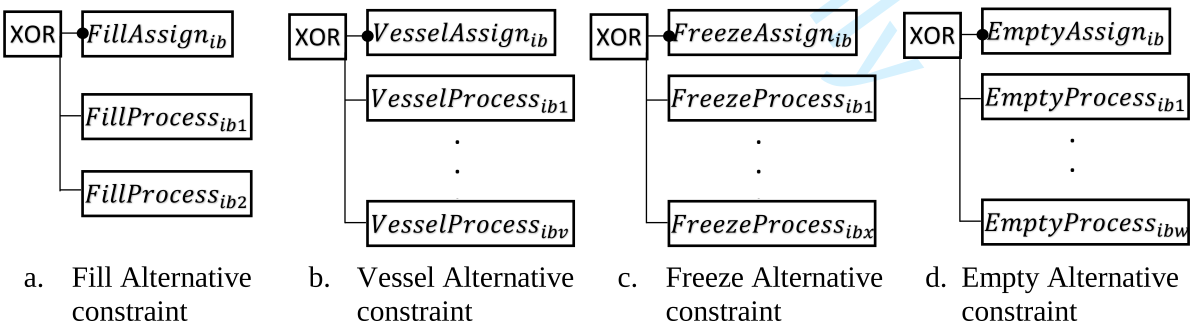


Figure 4. Alternative interval decision variables

Sequence decision variables arrange the processing order of product mixes for each equipment. Figure 5 shows the process interval assigned to the sequence variable. It is important to note that *Vesselprocess* combines the filling (pasteurization), aging, waiting, and emptying (freezing + packaging) steps.

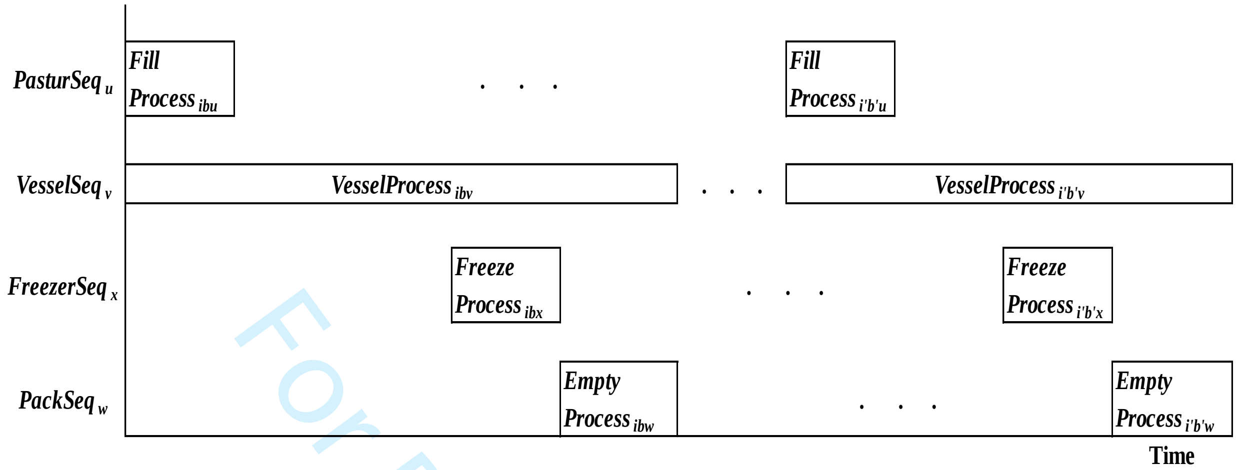


Figure 5. Sequence decision variables

Objective

The model minimizes a makespan objective. It is computed as the maximum of all batches' completion time for the vessel variables (*VesselAssign*).

$$Minimize \max_{i,b}\ endOf\left(VesselAssign_{i,b}\right)\ \ \forall i,\ b\ \ \text{where } i \in Product\ Mix,\ b \in 1..MinNoBatch_i \qquad (1)$$

Sequence constraints

Sequence constraints in the model prohibit the overlap of intervals (defined in sequence decision variables) in all machines. The constraints for pasteurizing, aging, freezing, and packaging stages are given in equations 2, 3, 4, and 5 respectively. These constraints also insert the respective changeover times between the processing intervals of consecutive batches. The changeover times required between product mixes are defined by two matrices given in Appendix 1c and 2c ($ProcessChOTimes_{ii'}$ for pasteurization and aging processes, and $PackageChOTimes_{ii'}$ for packaging process). The last digit enforces the changeover time added by each product batch to the rest of product mixes following it.

$$noOverlap\left(PasturSeq_u, ProcessChOTime_{i,\,i'}, 1\right) \quad \forall u \text{ where } u \in Pasteurizer,\ i\,\&\,i' \in Product\ Mix \tag{2}$$

$$noOverlap\left(VesselSeq_v, ProcessChOTime_{i,\,i'}, 1\right) \quad \forall v \text{ where } v \in Vessels,\ i\,\&\,i' \in Product\ Mix \tag{3}$$

$$noOverlap\left(FreezerSeq_x, ProcessChOTime_{i,\,i'}, 1\right) \quad \forall x \text{ where } x \in Freezers,\ i\,\&\,i' \in Product\ Mix \tag{4}$$

$$noOverlap\left(PackSeq_w, ProcessChOTime_{i,\,i'}, 1\right) \quad \forall w \text{ where } w \in Packaging\ Lines,\ i\,\&\,i' \in Product\ Mix \tag{5}$$

Interval constraints

Three groups of interval constraints are formulated for the optimization model. The first group constitutes constraints that interlink the start and end of intervals for successive stages of each product to create a processing chain (equations 6, 7, 8 and 9).

Pasteurization is immediately followed by the aging process (equation 6). Filling rate of the vessels are assumed to be equal to the pasteurization speed.

$$startAtEnd\left(AgeProcessV_{i,b}, FillAssign_{i,b}\right) \forall i,b \text{ where } i \in Product\ Mix,\ b \in\ 1..MinNoBatch_i \tag{6}$$

Equations 7, 8, and 9 connect the aging, waiting, freezing, and emptying. Right after the completion of the aging interval, the waiting interval commences (equation 7). Following this, equation 8 links the freezing stage with the waiting interval and finally equation 9 connects the freezing stage with the packaging stage.

$$startAtEnd\left(WaitingProcessV_{i,b}, AgeProcessV_{i,b}\right) \quad \forall i,b \text{ where } i \in Product\ Mix,\ b \in\ 1..MinNoBatch_i \tag{7}$$

$$startAtEnd\left(FreezeAssign_{i,b}, WaitingProcessV_{i,b}\right) \quad \forall i,b \text{ where } i \in Product\ Mix,\ b \in\ 1..MinNoBatch_i \tag{8}$$

$$startAtEnd\left(EmptyAssign_{i,b}, FreezeAssign_{i,b}\right) \quad \forall i,b \text{ where } i \in Product\ Mix,\ b \in\ 1..MinNoBatch_i \tag{9}$$

The processing in the aging vessels contains five distinct steps. These steps are 1) filling up the tanks; 2) aging; 3) waiting; 4) freezing; and 5) emptying the tanks. All these steps have to be completed before any succeeding batch can start its processing in the same vessel. An aggregating interval (*VesselAssign*) is created to integrate these steps by aligning it with the start and end of these processing steps. Explicitly, the starts of *VesselAssign* and *FillAssign* are aligned (equation 10) whereas the ends of *VesselAssign* and *EmptyAssign* are aligned (equation 11).

$$startAtEnd\left(FillAssign_{i,b}, VesselAssign_{i,b}\right) \quad \forall i,b \text{ where } i \in Product\ Mix,\ b \in 1..MinNoBatch_i \tag{10}$$

$$startAtEnd\left(VesselAssign_{i,b}, EmptyAssign_{i,b}\right) \quad \forall i,b \text{ where } i \in Product\ Mix,\ b \in 1..MinNoBatch_i \tag{11}$$

As described in the previous sections, multiple machines are allocated to process each product batch. However, only one of these processing alternatives can be inserted in the final schedule. The third group of interval constraints selects and assigns processing intervals to the schedule (equations 12, 13, 14, and 15). *FillAssign, VesselAssign, FreezeAssign*, and *EmptyAssign* variables represent the scheduled intervals for pasteurizing, aging, freezing and packaging stages respectively.

$$alternative\left(FillAssign_{i,b}, FillProcess_{i,b,u}\right) \quad \forall i,b \text{ where } i \in Product\ Mix,\ b \in 1..MinNoBatch_i, u \in Pasteurizer \tag{12}$$

$$alternative\left(VesselAssign_{i,b}, VesselProcess_{i,b,v}\right) \quad \forall i,b \text{ where } i \in Product\ Mix,\ b \in 1..MinNoBatch_i, v \in Vessels \tag{13}$$

$$alternative\left(FreezeAssign_{i,b}, FreezeProcess_{i,b,x}\right) \quad \forall i,b \text{ where } i \in Product\ Mix,\ b \in 1..MinNoBatch_i, x \in Freezers \tag{14}$$

$$alternative\left(EmptyAssign_{i,b}, EmptyProcess_{i,b,w}\right) \quad \forall i,b \text{ where } i \in Product\ Mix,\ b \in 1..MinNoBatch_i, w \in Packaging\ Line \tag{15}$$

Weekend break constraints

In the original problem, production takes place only during the weekdays in the facility. Therefore, the solution model cannot schedule any processing over the weekends and would have to cease production making proper break-up arrangements. The first part of this arrangement is to stop the production process earlier so that shut-down procedures would be completed before the end of the production week. This is expressed as a changeover to the idle state (given as 2 hours) which would effectively reduce the production week from 120 to 118 hours. Any interval for pasteurization, freezing and packaging processes cannot extend over the hours 118 to 120. A '*forbidExtent*' constraint is formulated to embody this scheduling restriction (equations 16, 17 and 18). A step function (*WeekendBreak*) defines these forbidden periods for a given production horizon.

$$forbidExtent\left(FillAssign_{i,b}, WeekendBreak\right) \quad \forall i,b \text{ where } i \in Product\ Mix,\ b \in 1..MinNoBatch_i \tag{16}$$

$$forbidExtent\left(FreezeAssign_{i,b}, WeekendBreak\right) \quad \forall i,b \text{ where } i \in Product\ Mix,\ b \in 1..MinNoBatch_i \tag{17}$$

$$forbidExtent\left(EmptyAssign_{i,b}, WeekendBreak\right) \quad \forall i,b \text{ where } i \in Product\ Mix,\ b \in 1..MinNoBatch_i \tag{18}$$

The weekend break arrangement for the aging process follows a different approach. Aging vessels act not only as processing machines but also as storage units. Hence any process started before the beginning of the weekend break can be finished and stored in the units over the weekends. The freezing/packaging process commences at the beginning of the subsequent production week. To represent this feature of the production process, a non-meta conditional constraint (case specific conditional constraint) is formulated (equation 19 and 20). Equation 19 assigns the normal aging time to the product when it is processed during the working week-days. Whenever the aging process extends over consecutive weeks, equation 20 assigns the time from the end of the fill process to the beginning of the later week as the aging time to product. As a result, the start of the freezing process coincides with the beginning of the week. Aging process cannot start during the weekends, and this constraint is enforced by equation 21.

$$\begin{aligned}
& endOf\left(FillAssign_{i,b}\right) \geq (l-1) * Week\ AND \\
& \quad endOf\left(FillAssign_{i,b}\right) + AgingTime_i \geq (l-1) * Week \text{ AND} \\
& \quad endOf\left(FillAssign_{i,b}\right) < (l * Week) - Idle\ AND \\
& \quad endOf\left(FillAssign_{i,b}\right) + AgingTime_i < (l * Week) - Idle => \\
& \quad sizeOf\left(FillAssign_{i,b}\right) = AgingTime_i \\
& \quad \forall i, b, l \text{ where } i \in Product\ Mix,\ b \in\ 1..MinNoBatch_i, l \in N
\end{aligned} \tag{19}$$

$$\begin{aligned}
& endOf\left(FillAssign_{i,b}\right) \geq (l-1) * Week\ AND \\
& \quad endOf\left(FillAssign_{i,b}\right) + AgingTime_i \geq l * Week \text{ AND} \\
& \quad endOf\left(FillAssign_{i,b}\right) < (l * Week) - Idle\ AND \\
& \quad endOf\left(FillAssign_{i,b}\right) + AgingTime_i < ((l+1) * Week) - Idle => \\
& \quad sizeOf\left(FillAssign_{i,b}\right) = (l * Week) - endOf\left(FillAssign_{i,b}\right) \\
& \quad \forall i, b, l \text{ where } i \in Product\ Mix,\ b \in\ 1..MinNoBatch_i, l \in 1..n-1
\end{aligned} \tag{20}$$

$$\begin{aligned}
& forbidStart\left(AgeProcessV_{i,b}, WeekendBreak\right) \\
& \qquad \forall i, b \text{ where } i \in Product\ Mix,\ b \in\ 1..MinNoBatch_i
\end{aligned} \tag{21}$$

Packaging line constraint

Processing in each packaging line would have to observe a precedence rule for product mix type. Higher priority product must be processed before lower priority products (in descending priority: M-L-K-J-I-H-G-F-E-D-C-B-A). A built-in function ('*endBeforeStart*') constraint is formulated to enforce this processing condition as given in equation 22.

$$\begin{aligned}
& endBeforeStart\left(EmptyProcessV_{i',b',w}, EmptyProcessV_{i,b,w}\right) \\
& \qquad \forall i, i', b, b' \text{ where } i \,\&\, i' \in Product\ Mix,\ b \,\&\, b' \in\ 1..MinNoBatch_i, \\
& \qquad w \in Packaging\ Lines,\ ord\left(ProductMix, i\right) > ord\left(ProductMix, i'\right)
\end{aligned} \tag{22}$$

Processing time window constraint

As perishable products, ice creams require a quick processing chain. For the case study in this paper, the maximum processing time window for each batch is 72 hours. Equation 23 constrains the total processing time (from start to finish) for all batches to be within this production time window.

$$lengthOf\left(VesselAssign_{i,b}\right) < 72 \ \forall i,b \text{ where } i \in Product\ Mix,\ b \in 1..MinNoBatch_i \quad (23)$$

4 Experiment Result and Discussion

Two sets of experimental runs were used to test the proposed model. The first set compared the model's performance with a published model from the literature. A modified data set was used to evaluate the performance of published and the proposed CP models. The second experimental run focused on solving the scheduling problem for the full-scale ice cream processing facility. This experiment aimed at finding the maximum number of product mixes that can be scheduled with the proposed model (disregarding limitations due to computing capacity and optimization software). For both sets of experiments, CPLEX v12.8 was used to formulate and execute the proposed model. All computations were performed on an Alienware Workstation with Intel® Core™ i7-5820K Processor CPU, 1TB HSD/3.6TB HDD, and 32GB RAM, on Windows 10 Enterprise operating systems.

4.1 Experimental Run 1

The publication by Wari and Zhu (2016) was selected as the base for comparison of the performance of the proposed CP model in this paper. This study presented an MILP model to schedule the partial production processing the ice cream facility. It formulated sets and subsets, parameters and parameter functions, decision variables, and objective and constraint functions to represent the production environment. Sets, subsets, parameters, and parameter functions have been created to define product-mix type, available processing equipment for each stage, processing parameter such as processing time, and setup time, scheduling parameter (including time horizon). Decision variables consist of the start and completion of processing times, waiting times, a binary variable for product processing order, and a binary variable for selection of aging vessel. A makespan objective was used. Four groups of constraints were formulated to solve the problem. The first group created process intervals by defining the start and completion time for each product batch. The second group of constraints focused on the allocation of batches to vessels. These constraints assigned each batch to a vessel in a cyclic manner. The next group rearranged product batches assigned to an equipment based on predefined priority level. The last group consisted of miscellaneous constraints such as makespan, lower bound and domain sets for decision variables (Wari & Zhu, 2016).

Due to the limitation of each model, the test data used by Wari and Zhu was modified in this paper to compare the two models. Since the CP model can be formulated as integer model, all parameters and parameter function were converted to integer values for the MILP model too. The problem model for the MILP considered only one pasteurization unit and no freezer stage. To accommodate this reduction in model size, the CP model considered only one pasteurization equipment. Furthermore, the freezing stage was excluded from the original model by relaxing all constraints relating to the stage (constraints 4, 8, 14, and 17) and modifying constraint 9. To test the performance of each solution model (MILP and CP) over longer production horizon, batch sizes for problem instances were stretched from 180 or 200 to 640 batches for all three problem sets. In other words, the modified problem model mixed up part of the data from the published paper and new problem instance with larger batch sizes. The new problem instances were created by a random generation of demand size for each product mix. The complete list of these values is given in Appendix 1. Finally, the run parameter limits for CP model were set 600 seconds maximum run time, similar to the limit set in Wari and Zhu (2016).

Table 2 compared the run results of the MILP and CP models. The table showed the makespan results for the three test demand sets (Appendix 1a) and their respective computation time for each problem instances. The last column gave the makespan difference between the two models. Overall, the difference ranged from -3% to -15% for most problem instants. For a maximum of 10 minutes of run time, the CP model could not achieve the optimal value. For small size problem instances, the MILP model attained optimal value faster than the CP model. As the size of these instances increased, the runtime needed for MILP to solve the problem increased sharply than the time needed for the CP model. The MILP model failed to obtain any results for large problems given the time limit while the CP model was able to obtain results in all the instances.

Table 2 Run result summary for experiment 1 – for modified problem instances from Wari and Zhu (2016)

Set 1 (8 product mixes)						
Problem Instances	Total Batch Size	MILP model Make span (in hrs)	MILP model Run time (in s)	CP model Make span (in hrs)	CP model Run time (in s)	Makespan Result Variation (hrs)
1	80	129	15	138	600	-7%
2	100	161	29	173	600	-7%
3	120	191	44	203	600	-6%
4	140	228	58	245	600	-7%
5	160	254	92	271	600	-7%
6	180	295	126	308	600	-4%
7	200	318	141	345	600	-9%
8	220	336	167	378	600	-13%
9	240	381	187	404	600	-6%
10	260	429	221	442	600	-3%
11	280	456	236	482	600	-6%
12	320	512	339	544	600	-6%
13	360	569	588	612	600	-8%
14	400	664	528	697	600	-5%
15	440	-	600	745	600	-
16	480	-	600	814	600	-
17	520	-	600	884	600	-
18	560	-	600	954	600	-
19	600	-	600	1022	600	-
20	640	-	600	1090	600	-

Set 2 (16 product mixes)						
Problem Instances	Total Batch Size	MILP model Make span (in hrs)	MILP model Run time (in s)	CP model Make span (in hrs)	CP model Run time (in s)	Makespan Result Variation (hrs)
21	80	129	19	144	600	-12%
22	100	162	40	182	600	-12%
23	120	192	75	214	600	-12%
24	140	224	144	247	600	-10%
25	160	249	188	280	600	-12%
26	180	296	512	317	600	-7%
27	200	318	600	352	600	-11%
28	220	348	600	388	600	-11%
29	240	474	600	420	600	11%
30	260	-	600	455	600	-
31	280	-	600	498	600	-
32	320	-	600	561	600	-
33	360	-	600	633	600	-
34	400	-	600	700	600	-
35	440	-	600	770	600	-
36	480	-	600	838	600	-
37	520	-	600	906	600	-
38	560	-	600	990	600	-
39	600	-	600	1055	600	-
40	640	-	600	1124	600	-

Problem Instances	Total Batch Size	MILP model Make span (in hrs)	MILP model Run time (in s)	CP model Make span (in hrs)	CP model Run time (in s)	Makespan Result Variation (hrs)
Set 3 (24 product mixes)						
41	80	140	18	151	600	-8%
42	100	164	37	180	600	-10%
43	120	193	63	213	600	-10%
44	140	223	99	250	600	-12%
45	160	258	185	284	600	-10%
46	180	285	301	322	600	-13%
47	200	334	298	357	600	-7%
48	220	355	600	391	600	-10%
49	240	387	600	417	600	-8%
50	260	-	600	453	600	-
51	280	-	600	505	600	-
52	320	-	600	577	600	-
53	360	-	600	645	600	-
54	400	-	600	699	600	-
55	440	-	600	772	600	-
56	480	-	600	863	600	-
57	520	-	600	910	600	-
58	560	-	600	995	600	-
59	600	-	600	1060	600	-
60	640	-	600	1141	600	-

Overall, the proposed model provided more flexibility for schedulers by relaxing the constraints to assign product mixes before optimizing the production. Batches of different products can be arranged freely without the need for prioritization of products. However, it made the CP model more complex. Evident from the results, the proposed model attained good schedules for longer production horizons where the MILP model failed to attain any.

4.2 Experimental Run 2

The second experimental run tested the performance of the proposed CP model to optimize production schedules for the full capacity of the facility considering all the stages and equipment. The pasteurization process is synchronized with the filling step of the aging vessels. After completing aging (a discrete process), product mixes are continuously emptied from the vessel through the freezers and packaging lines as a continuous process to complete the production. The complete list of processing speed or time for each product and equipment are given in Appendix 2. It is important to note that the processing speed for the freezing and packaging processes vary and the proposed model assumes a speed buffering mechanism (such as temporary storage) exists between these two stages.

Because no large problem data can be found in the literature, we developed two sets of demand data to test the proposed model. Two batch sizes were assumed for the vessels in the facility: 4,000 and 8,000 Kg. Each product mix takes only one of these batch sizes, and multiple batches were used to generate demand volume for the mix. The first demand set predominantly consists of the 4K-batch of product mixes. The total batch size for this set started at 40 and grew to 400 batches for the last problem instance. In demand set 2, 8K dominated total demand. The two demand sets and additional parameters for the model are given in Appendix 2. The specific set of machines in which each product mix complete its processing were also given in this appendix.

The resulting schedules for the two demand sets extended over 4 to 5 production weeks, as shown in Table 3. For the same total batch sizes, the model attained varying makespan results for two demand sets. It can also be observed that the model obtained feasible solutions for more problem instances in set 1 than set 2. The cause for these variations could be associated with the variation of processing time between product mixes of the two batch sizes and the number of available processing machines (fewer number of vessels, and packaging lines for 8K). Like the first experiment, the proposed model could not obtain optimal values within the allowed 10 minutes runtime for any of the demand sets. The two test sets also showed maximum problem sizes that the model can handle. The model solved problem instants with batch sizes up to 360 for set 1 (39000 constraints and 7000 variable) whereas it attained results for batch sizes 240 for set 2 (17000 constraints and 4000 variable). To provide some insights on the optimization size of each problem instant, the last column of each set in Table 3 gave the number of constraints and variables generated to solve the problem by the IBM ILOG CP.

Table 3 Run result summary for experiment 2

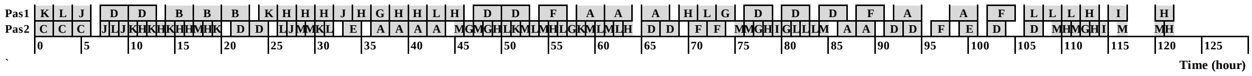
a. Set 1

Problem Instance	Total Batch	CP model		Number of constraints/ variables
		Make span (in hrs)	Run time (in s)	
1	40	40	600	1719/857
2	80	72	600	4604/1762
3	120	127	600	7440/2530
4	160	233	600	12929/3556
5	200	233	600	15674/4131
6	240	286	600	23647/5156
7	280	331	600	27807/5682
8	320	396	600	31607/6199
9	360	678	600	37289/6945
10	400	No solution	600	54543/8209

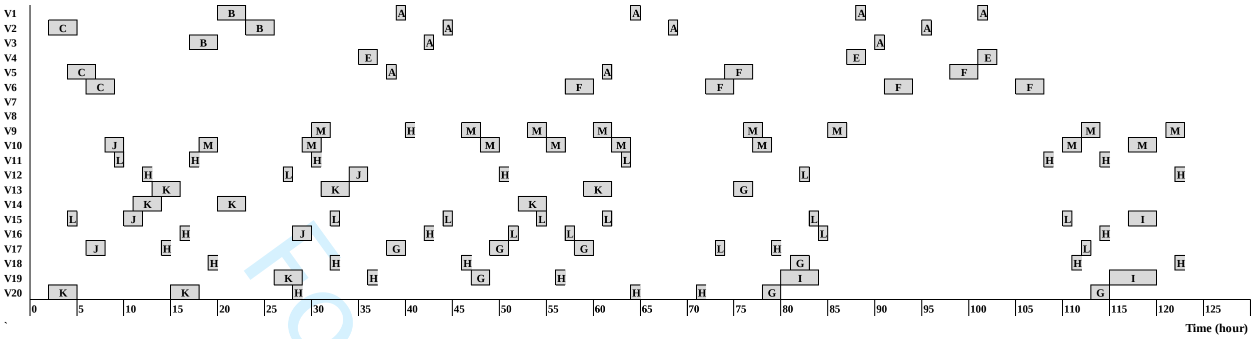
b. Set 2

Problem Instance	Total Batch	CP model		Number of constraints/ variables
		Make span (in hrs)	Run time (in s)	
11	40	70	600	1569/772
12	80	139	600	3745/1503
13	120	318	600	5519/2177
14	160	357	600	9253/3020
15	200	357	600	12673/3822
16	240	460	600	17480/4529
17	280	No solution	600	22010/5254
18	320	No solution	600	25045/5958
19	360	No solution	600	39113/7242
20	400	No solution	600	40197/7645

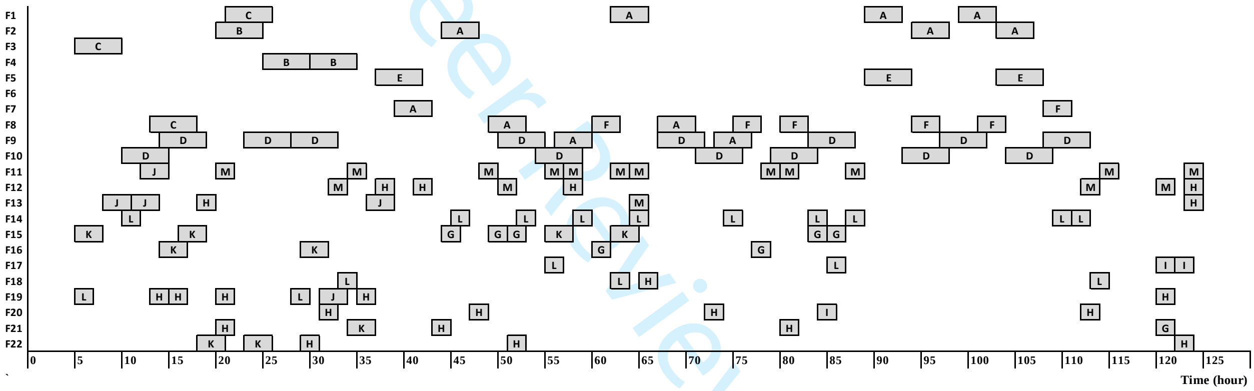
Figure 6 presented the schedule for problem instance 3 (makespan value of 127 hours). The schedule showed the bottleneck machine to be the pasteurizing units, unlike the cases in experiment 1 in which the packaging units were the bottleneck (Wari & Zhu, 2016). For the remaining stages, under-utilization of most of the machines was observed. The increased number of available machines gave product mixes additional alternatives to complete their processing. The figure also showed the weekend breaks for the pasteurization, freezer and packaging line processing stage where production is halted and restarted on the following week. Some aging vessels (specifically vessel 10, 15 and 19) stored products over the weekends before the production commenced the freezing and packaging processes at the beginning of the second week.



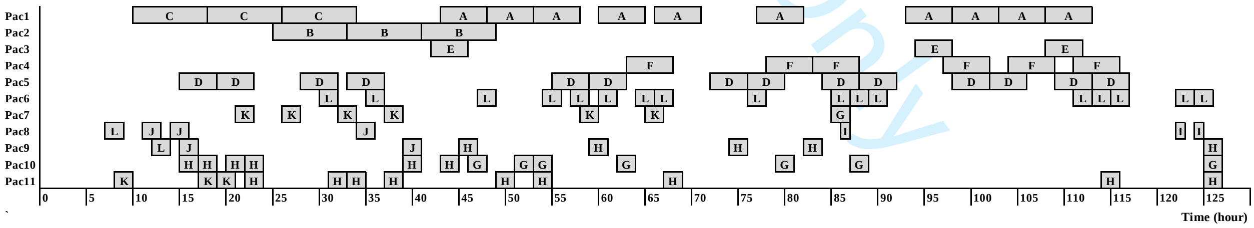
a. Pasteurizers schedules: Items A to M are product mix types



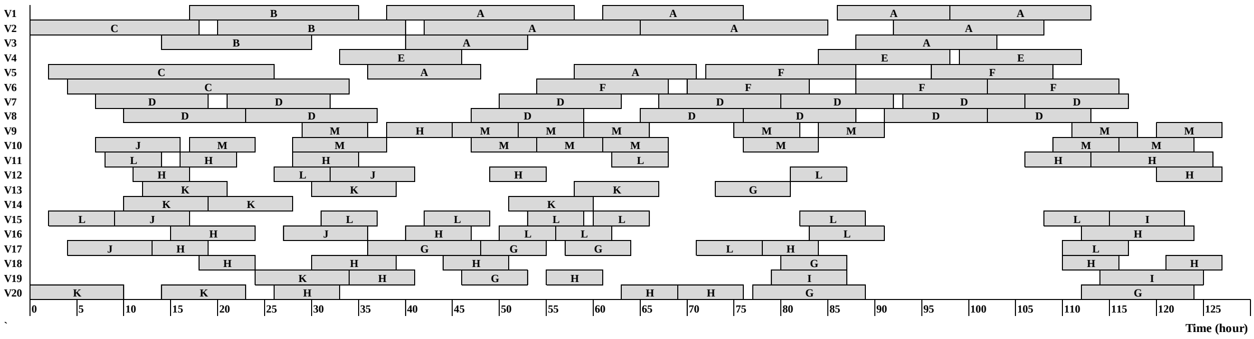
b. Aging tanks schedules: Items A to M are product mix types



c. Freezer schedules: Items A to M are product mix types



d. Packaging lines schedules: Items A to M are product mix types



e. Vessel schedules: Items A to M are product mix types

Figure 6. The production schedule for problem instance 3

5 Conclusion

The proposed CP model in this paper solved a large scheduling problem previously considered too complex and large due to optimization method limitations. Processing interval for each product mix and sequencing of these intervals in their respective machines were formulated as variables in the model. Constraints to prevent overlapping of processing interval, link processing stage, order processing stages, and restrict production only to the weekdays were integrated to emulate some limitations of the production environment. For a makespan objective, the model was compared with an MILP model using a modified larger demand data from the literature. The comparison showed that the model attained a makespan value comparable (about 15% deviation) to the MILP model. However, for long scheduling horizon, it showed better performance by solving all problem instance where the MILP failed to attain results. For the primary challenge of this paper, the model was applied to a scheduling problem for multi-stage ice cream processing with multiple machines in each stage for a full-size production facility. Two demand data sets were developed to test the performance of the model. The CP model reported a respective schedule extending over multiple production weeks. It also showed that the model could not solve large product mixes for both sets.

The proposed CP models can optimize more complex scheduling problems and attain better results within a relatively short run time limit. It also gives practitioners the option of choosing between “optimal result but small-size problem capacity” and “good results but large-size problem capacity" approaches. The method could easily be integrated into an existing production system for a company. Future research directions could explore technical and financial aspects of this integration. Other optimization objectives, such as Tardiness and Earliness, are also promising directions to extend research in this area. It would also be interesting to see the application of CP scheduling in other food processing industries (other processing industries) with even more manufacturing constraints. Scheduling problems could be combined with other production function such as inventory, transportation and distribution, and production planning. The performance of CP approach in these areas could also be a topic of interest.

The CP model in this paper obtained results within a short optimization runtime for large problems. This could be used to provide timely inputs for decision making in production planning and scheduling activities of any company. Such inputs can give companies a competitive edge in the highly dynamic market currently observed in the industry.

References

Apt, K. (2003). *Principles of constraint programming.* Cambridge University Press.

Banerjee, S., Dangayach, G. S., Mukherjee, S. K., & Mohanti, P. K. (2008). Modelling process and supply chain scheduling using hybrid meta-heuristics. *Metaheuristics for scheduling in industrial and manufacturing applications*, 277-300.

Bockmayr, A., & Hooker, J. N. (2005). Constraint programming. In G. N. K. Aardal, *Handbooks in Operations Research and Management Science* (pp. 559-600).

Bongers, P. M., & Bakker, B. H. (2006). Application of multi-stage scheduling. *16th European Symposium on Computer Aided Process Engineering and 9th International Symposium on Process Systems Engineering* (pp. 1917–1922). Boston: Elsevier.

Doganis, P., & Sarimveis, H. (2007). Optimal scheduling in a yogurt production line based on mixed integer linear programming. *Journal of Food Engineering*, 445-453.

Doganis, P., & Sarimveis, H. (2008a). Mixed Integer Linear Programming Scheduling in the Food Industry. In D.-W. Sun, & F. Erdogdu, *Optimization in food engineering* (p. 305). New York: Taylor & Francis Group, LLC.

Doganis, P., & Sarimveis, H. (2008b). Optimal production scheduling for the dairy industry . *Annals of Operations Research*, 315-331.

El Hachemi, N., Gendreau, M., & Rousseau, L. (2011). A hybrid constraint programming approach to the log-truck scheduling problem. *Annals of Operations Research*, 163-178.

Gellert, T., Höhn, W., & Möhring, R. H. (2011). Sequencing and scheduling for filling lines in dairy production. *Optimization Letters* , 491-504.

Goel, V., Slusky, M., van Hoeve, W., Furman, K., & Shao, Y. (2015). Constraint programming for LNG ship scheduling and inventory management. *European Journal of Operational Research*, 662-673.

Han, A., & Li, E. (2014). A constraint programming-based approach to the crew scheduling problem of the Taipei mass rapid transit system. *Annals of Operations Research*, 173-193.

Harjunkoski, I., Maravelias, C., Bongers, P., Castro, P., Engell, S., Grossmann, I., . . . Wassick, J. (2014). Scope for industrial applications of production scheduling models and solution methods. *Computers & Chemical Engineering*, 161-93.

Hecker, F. T., Hussein, W. B., Paquet-Durand, O., Hussein, M. A., & Becker, T. (2013). A case study on using evolutionary algorithms to optimize bakery production planning. *Expert Systems with Applications*, 6837-6847.

Hecker, F. T., Stanke, M., Becker, T., & Hitzmann, B. (2014). Application of a modified GA, ACO and a random search procedure to solve the production scheduling of a case study bakery. *Expert Systems with Applications*, 5882-5891.

Heinonen, J., & Pettersson, F. (2003). Scheduling a specific type of batch process with evolutionary computation. *Evolutionary Computation* (pp. 966-970). IEEE.

I., L., & JF., P. (2013). Constraint Programming. In G. S.I., & F. M.C., *Encyclopedia of Operations Research and Management Science* (pp. 267-273). Boston, MA: Springer.

IBM. (2016). IBM ILOG CPLEX Optimization Studio CP Optimizer User's Manual. *Version 12 Release 7*. Armonk, NY, USA: IBM Corp.

Jolai, F., Rabiee, M., & Asefi, H. (2012). A novel hybrid meta-heuristic algorithm for a no-wait flexible flow shop scheduling problem with sequence dependent setup times. *International Journal of Production Research*, 7447-66.

Karray, A., Benrejeb, M., & Borne, P. (2011). New parallel genetic algorithms for the single-machine scheduling problems in agro-food industry. *Communications, Computing and Control Applications (CCCA)* (pp. 1-7). Hammamet: IEEE.

Kopanos, G. M., Puigjaner, L., & Georgiadis, M. C. (2009). Optimal production scheduling and lot-sizing in dairy plants: the yogurt production line. *Industrial & Engineering Chemistry Research*, 701-718.

Kopanos, G. M., Puigjaner, L., & Georgiadis, M. C. (2011). Production scheduling in multiproduct multistage semicontinuous food processes. *Industrial & Engineering Chemistry Research*, 6316-6324.

Kopanos, G. M., Puigjaner, L., & Georgiadis, M. C. (2012). Efficient mathematical frameworks for detailed production scheduling in food processing industries. *Computers & Chemical Engineering*, 206-216.

Laborie, P., & Rogerie, J. (2008). Reasoning with Conditional Time-Intervals. *FLAIRS conference*, (pp. 555-560).

Laborie, P., Rogerie, J., Shaw, P., & Vilím, P. (2009). Reasoning with Conditional Time-Intervals. Part II: An Algebraical Model for Resources. *FLAIRS conference* (pp. 201-206). Florida, USA: AAAI Publications.

Liu, S., Pinto, J. M., & Papageorgiou, L. G. (2010). Single-stage scheduling of multiproduct batch plants: An edible-oil deodorizer case study. *Industrial & Engineering Chemistry Research*, 8657-8669.

Moradinasab, N., Shafaei, R., Rabiee, M., & Ramezani, P. (2013). No-wait two stage hybrid flow shop scheduling with genetic and adaptive imperialist competitive algorithms. *Journal of Experimental & Theoretical Artificial Intelligence*, 207-25.

Nagano, M. S., Miyata, H. H., & Araújo, D. C. (2015). A constructive heuristic for total flowtime minimization in a no-wait flowshop with sequence-dependent setup times. *Journal of Manufacturing Systems, 36*, 224-230. Retrieved 11 22, 2017, from http://sciencedirect.com/science/article/pii/s0278612514000739

Novas, J., & Henning, G. (2012). A comprehensive constraint programming approach for the rolling horizon-based scheduling of automated wet-etch stations. *Computers & Chemical Engineering*, 189-205.

Öztürk, C., Tunalı, S., Hnich, B., & Örnek, M. (2012). A Constraint Programming Model for Balancing and Scheduling of Flexible Mixed Model Assembly Lines with Parallel Stations. *IFAC Proceedings*, (pp. 420-425.). Bucharest, Romania.

Rossi, F., Van Beek, P., & Walsh, T. (2006). *Handbook of constraint programming.* Elsevier.

Sadi-Nezhad, S., & Darian, S. B. (2010). Production scheduling for products on different machines with setup costs and times. *International Journal of Engineering and Technology*, 410-418.

Samarghandi, H., & ElMekkawy, T. Y. (2012). A meta-heuristic approach for solving the no-wait flow-shop problem. *International Journal of Production Research* , 7313-7326.

Sel, Ç., Bilgen, B., & Bloemhof-Ruwaard, J. (2017). Planning and scheduling of the make-and-pack dairy production under lifetime uncertainty. *Applied Mathematical Modelling*, 129-144.

Shaw, K. J., Lee, P. L., Nott, H. P., & Thompson, M. (2000). Genetic algorithms for multiobjective scheduling of combined batch/continuous process plants. *Evolutionary Computation* (pp. 293-300). La Jolla, CA: IEEE.

Topaloglu, S., & Ozkarahan, I. (2011). A constraint programming-based solution approach for medical resident scheduling problems. *Computers & Operations Research*, 246-255.

Unsal, O., & Oguz, C. (2013). Constraint programming approach to quay crane scheduling problem. *Transportation Research Part E*, 108-122.

Wang, S., & Liu, M. (2013). A genetic algorithm for two-stage no-wait hybrid flow shop scheduling problem. *Computers & Operations Research*, 1064-1075.

Wang, S., Liu, M., & Chu, C. (2015). A branch-and-bound algorithm for two-stage no-wait hybrid flow-shop scheduling. *International Journal of Production Research*, 1143-1167.

Wari, E., & Zhu, W. (2016). Multi-week MILP scheduling for an ice cream processing facility. *Computers & Chemical Engineering*, 141-156.

Wauters, T., Verbeeck, K., Verstraete, P., Berghe, G. V., & De Causmaecker, P. (2012). Real-world production scheduling for the food industry: An integrated approach. *Engineering Applications of Artificial Intelligence*, 222-228.

Ye, H., Li, W., & Abedini, A. (2017). An improved heuristic for no-wait flow shop to minimize makespan. *Journal of Manufacturing Systems, 44*, 273-279. Retrieved 11 22, 2017, from http://sciencedirect.com/science/article/pii/s0278612517300468

Ye, H., Li, W., & Miao, E. (2016). An effective heuristic for no-wait flow shop production to minimize makespan. *Journal of Manufacturing Systems*, 2-7.

Zeballos, L., Quiroga, O., & Henning, G. (2010). A constraint programming model for the scheduling of flexible manufacturing systems with machine and tool limitations. *Engineering Applications of Artificial Intelligence*, 229-248.

Zhang, L., & Wong, T. (2012). 2012. Solving integrated process planning and scheduling problem with constraint programming. *In Proceedings of the Asia Pacific Industrial Engineering & Management Systems Conference* (pp. 1525-1532). Phuket, Thailand: APIEMS.

Zhou, Y., & Gu, X. (2009). Research on no-wait flow shop scheduling problem with fuzzy due date based on evolution games. *Computer Science and Information Technology* (pp. 495-499). Beijing: IEEE.

Appendix

Appendix 1: Experiment 1 data

a. Modified problem instant data for three set types from Wari & Zhu (2016) (The numbers are in 1,000Kg)

Product Mix	Problem Instance (Set 1)																			
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
A	48	96	32	48	80	96	200	208	128	144	192	160	160	280	176	304	240	360	352	368
B	16	16	80	96	16	64	120	88	32	80	80	32	112	168	240	208	304	232	336	352
C	64	72	32	64	80	128	88	32	144	192	200	160	176	264	224	216	264	264	240	328
D	32	24	112	96	160	88	40	176	192	120	112	320	304	144	296	304	312	344	368	328
E	48	68	124	124	120	252	160	132	168	300	320	240	292	440	500	272	440	452	524	528
F	32	40	16	176	68	104	60	88	100	136	144	136	116	332	296	428	568	176	304	640
G	80	76	144	92	120	52	248	136	200	132	128	240	344	168	288	368	352	592	524	392
H	80	112	68	16	164	124	108	272	244	204	236	328	312	232	208	336	160	420	400	312
Batch Total	80	100	120	140	160	180	200	220	240	260	280	320	360	400	440	480	520	560	600	640

Product Mix	Problem Instance (Set 2)																			
	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
A	56	64	48	16	32	64	128	80	96	136	64	80	96	176	216	176	104	200	176	176
B	16	8	48	80	88	16	16	56	96	48	104	96	136	72	120	112	152	72	152	248
C	8	16	32	56	40	48	32	64	64	88	80	104	120	96	80	64	120	88	144	176
D	24	16	8	56	16	64	32	48	16	40	104	120	104	88	72	144	112	192	160	208
I	12	68	40	100	8	152	136	88	80	84	56	252	188	260	156	92	136	260	284	372
J	20	24	88	20	36	20	48	96	176	124	72	40	116	132	104	176	188	112	204	152
K	44	20	12	136	28	60	40	60	24	68	116	196	196	92	112	156	156	180	160	388
L	32	8	12	20	84	60	16	108	24	160	100	80	128	80	132	80	172	104	116	136
E	16	16	64	16	24	16	32	48	128	32	64	32	64	96	72	216	216	168	160	128
F	8	24	24	8	112	48	48	56	48	40	104	56	64	96	168	72	144	112	192	120
G	24	32	16	56	8	128	64	80	32	104	48	184	136	176	96	136	152	256	192	224
H	40	72	32	24	16	64	144	80	64	120	48	88	104	168	200	224	216	288	240	200
M	20	4	16	20	176	28	8	60	32	36	120	48	80	48	204	176	164	172	100	148
N	8	8	16	68	12	20	16	80	32	80	124	88	148	44	36	112	136	116	192	232
O	60	104	104	20	68	140	208	64	208	108	156	160	84	348	336	380	328	460	412	224
P	28	40	56	20	60	16	80	68	112	76	68	36	88	112	168	176	192	148	224	168
Batch Total	80	100	120	140	160	180	200	220	240	260	280	320	360	400	440	480	520	560	600	640

Product Mix	Problem Instance (Set 3)																			
	41	**42**	**43**	**44**	**45**	**46**	**47**	**48**	**49**	**50**	**51**	**52**	**53**	**54**	**55**	**56**	**57**	**58**	**59**	**60**
A	24	24	16	56	32	40	56	40	80	72	64	64	88	64	112	136	128	144	144	152
B	16	8	40	40	32	8	56	48	48	80	72	64	88	72	88	128	120	104	120	104
C	24	40	8	16	64	16	56	48	56	24	56	128	120	72	40	136	128	136	104	176
D	8	16	48	48	32	64	80	64	64	96	64	64	112	144	160	168	144	152	192	192
I	24	8	8	24	56	24	8	16	32	32	56	112	64	64	56	40	96	152	112	96
J	24	8	8	16	16	64	16	16	24	24	56	32	32	56	88	56	48	72	104	104
K	16	16	24	16	48	40	64	40	32	40	56	96	112	104	80	144	104	152	128	168
L	8	32	8	16	8	72	8	40	48	24	24	16	16	64	96	24	64	72	104	120
Q	8	8	16	16	16	8	16	24	24	32	32	32	32	56	40	40	56	72	56	48
R	16	8	56	40	8	16	40	64	48	96	88	16	48	104	112	96	112	88	120	72
S	16	40	16	16	16	16	16	56	56	32	48	32	32	56	48	48	88	80	64	88
T	8	16	24	16	40	80	64	40	32	40	40	80	104	72	120	136	96	88	160	200
E	20	16	36	40	60	88	136	52	56	76	76	120	196	132	164	292	152	180	224	300
F	4	36	28	40	8	4	8	64	76	68	36	16	16	60	72	20	112	80	80	56
G	12	12	20	40	44	8	64	32	52	60	44	88	108	64	68	140	116	136	112	128
H	20	36	32	20	4	84	8	68	56	52	72	8	12	96	136	36	92	56	140	132
M	40	12	24	40	8	4	4	36	52	64	104	16	12	104	68	48	84	128	76	28
N	12	12	12	40	148	136	80	24	52	52	36	296	228	100	188	172	212	276	336	376
O	20	20	4	20	36	8	40	24	40	24	44	72	76	68	32	100	80	120	68	104
P	8	8	4	40	12	20	28	12	48	44	20	24	40	96	64	64	64	108	76	68
U	24	36	52	20	12	68	80	88	56	72	100	24	92	100	140	184	120	124	152	196
V	4	12	4	40	8	16	8	16	52	44	12	16	16	68	60	20	64	92	68	44
W	52	36	40	40	80	16	80	76	76	80	144	160	160	104	96	212	196	184	176	212
X	8	52	88	20	36	44	24	140	72	108	104	72	60	144	152	56	196	100	188	156
Batch Total	80	100	120	140	160	180	200	220	240	260	280	320	360	400	440	480	520	560	600	640

b. Processing parameters data

	Aging Time	Min VesselSize	Filling rate	Empty rate	Filling Time	Empty Time
A	1	8000	4500	1750	2	5
B	3	8000	4500	1500	2	5
C	3	8000	4500	1000	2	8
D	0	8000	4500	1500	2	5
I	2	8000	4500	1750	2	5
J	3	8000	4500	1500	2	5
K	2	8000	4500	2000	2	4
L	1	8000	4500	2000	2	4
Q	4	8000	4500	2500	2	3
R	2	8000	4500	1250	2	6
S	3	8000	4500	1500	2	5
T	1	8000	4500	2250	2	4
E	2	4000	4500	1750	1	2
F	2	4000	4500	2000	1	2
G	2	4000	4500	2000	1	2
H	2	4000	4500	2000	1	2
M	3	4000	4500	2250	1	2
N	2	4000	4500	2000	1	2
O	3	4000	4500	1750	1	2
P	2	4000	4500	2250	1	2
U	1	4000	4500	1500	1	3
V	2	4000	4500	2000	1	2
W	2	4000	4500	1750	1	2
X	2	4000	4500	2750	1	1

c. Processing changeover times (in hours)

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
A	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
B	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
C	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
D	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
E	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
F	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
G	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
H	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
I	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
J	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
K	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1
L	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1
M	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
N	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
O	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
P	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
Q	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
R	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
S	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
T	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
U	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
V	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
W	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
X	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0

d. Packaging line changeover times (in hours)

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
A	0	1	1	1	1	1	1	1	1	1	1	1												
B	0	0	1	1	1	1	1	1	1	1	1	1												
C	0	0	0	1	1	1	1	1	1	1	1	1												
D	0	0	0	0	1	1	1	1	1	1	1	1												
E	0	0	0	0	0	1	1	1	1	1	1	1												
F	0	0	0	0	0	0	1	1	1	1	1	1												
G	0	0	0	0	0	0	0	1	1	1	1	1												
H	0	0	0	0	0	0	0	0	1	1	1	1												
I	0	0	0	0	0	0	0	0	0	1	1	1												
J	0	0	0	0	0	0	0	0	0	0	1	1												
K	0	0	0	0	0	0	0	0	0	0	0	1												
L	0	0	0	0	0	0	0	0	0	0	0	0												
M													0	1	1	1	1	1	1	1	1	1	1	1
N													0	0	1	1	1	1	1	1	1	1	1	1
O													0	0	0	1	1	1	1	1	1	1	1	1
P													0	0	0	0	1	1	1	1	1	1	1	1
Q													0	0	0	0	0	1	1	1	1	1	1	1
R													0	0	0	0	0	0	1	1	1	1	1	1
S													0	0	0	0	0	0	0	1	1	1	1	1
T													0	0	0	0	0	0	0	0	1	1	1	1
U													0	0	0	0	0	0	0	0	0	1	1	1
V													0	0	0	0	0	0	0	0	0	0	1	1
W													0	0	0	0	0	0	0	0	0	0	0	1
X													0	0	0	0	0	0	0	0	0	0	0	0

Appendix 2: Experiment 2 data

a. Problem instance data for two set types (The numbers are in 1,000Kg)

	Problem Instant set 1									
i	1	2	3	4	5	6	7	8	9	10
A	8	16	88	48	64	152	136	112	224	208
B	16	48	24	152	192	200	128	216	160	296
C	8	32	24	64	128	88	104	80	96	168
D	8	16	112	96	64	128	296	384	280	448
E	24	32	24	64	128	112	120	96	408	96
F	16	16	48	56	64	120	176	232	112	224
G	8	48	32	80	96	100	96	72	76	100
H	16	24	88	120	48	144	136	104	128	184
I	4	16	12	16	32	36	80	140	96	264
J	20	64	20	96	128	120	72	76	80	96
K	32	32	36	40	60	64	140	172	64	120
L	24	24	68	16	52	40	60	68	200	32
M	16	32	64	32	64	56	56	88	156	84
∑ Min NoBatch	40	80	120	160	200	240	280	320	360	400
8k	10	20	40	60	80	100	120	140	160	180
4k	30	60	80	100	120	140	160	180	200	220

	Problem Instant set 2									
i	1	2	3	4	5	6	7	8	9	10
A	32	96	80	120	136	168	160	112	472	248
B	8	48	120	72	160	152	168	248	168	408
C	48	88	96	96	112	96	112	320	272	112
D	64	80	56	184	232	120	288	168	304	176
E	16	48	192	216	216	272	392	408	208	304
F	72	120	96	112	104	312	160	184	176	512
G	4	8	8	48	16	72	96	80	92	60
H	12	12	12	16	24	48	60	72	108	132
I	4	8	20	64	52	96	100	48	96	84
J	4	8	32	32	120	32	40	84	68	112
K	8	24	64	8	60	24	44	120	84	184
L	4	12	8	56	32	80	88	60	104	92
M	4	8	16	16	16	48	52	96	88	56
∑ Min NoBatch	40	80	120	160	200	240	280	320	360	400
8k	30	60	80	100	120	140	160	180	200	220
4k	10	20	40	60	80	100	120	140	160	180

b. Processing parameters data (in Vessel size in kg and the rest in hours)

i	MinVessel Size	Fillingrate 1	Fillingrate 2	AgingTime	Freeingrate	EmptyTime	Idle
A	8000	3	2	1	4	5	2
B	8000	3	2	3	5	5	2
C	8000	3	2	3	5	8	2
D	8000	3	2	0	5	4	2
E	8000	3	2	2	5	4	2
F	8000	3	2	3	3	5	2
G	4000	2	1	2	2	2	2
H	4000	2	1	1	2	2	2
I	4000	2	1	4	2	1	2
J	4000	2	1	2	3	2	2
K	4000	2	1	3	3	2	2
L	4000	2	1	1	2	2	2
M	4000	2	1	2	2	2	2

c. Machine assignment for each product mix

	Pasturizer	Vessel	Freezer	Packaging
A	1, 2	1, 2, 3, 5, 6, 7, 8	1, 2, 7, 8, 9, 10	1, 4, 5
B	1	1, 2, 3	1, 2, 3, 4	1, 2, 5
C	1, 2	1, 2, 3, 5, 6	1, 2, 7, 8	1, 4
D	1, 2	7, 8	9, 10	5
E	1, 2	4	5, 6	3
F	1, 2	5, 6	7, 8	4
G	1, 2	20, 19, 18, 17, 16, 14, 13	22, 21, 20, 19, 18, 16, 15	8, 10 11, 12
H	1, 2	20, 19, 18, 17, 16, 12, 11, 10, 9	22, 21, 20, 19, 18, 14, 13, 11, 12	6, 7, 10, 11, 12
I	1, 2	20, 19, 18, 17, 16, 15	22, 21, 20, 19, 18, 17	9, 10, 11, 12
J	1, 2	17, 16, 15, 12, 11, 10, 9	19, 18, 17, 14, 13, 12, 11	10, 9, 7, 6
K	1, 2	20, 19, 14, 13	22, 21, 16, 15	12, 8
L	1, 2	17, 16, 15, 12, 11	19, 18, 17, 14, 13	10, 9, 7
M	2	10, 9	12, 11	6

d. Product specific processing rates

Product Code	Aging Period (hr)	Freezer Rate (Kg/hr)	Packaging Rate (Kg/hr)	Vessel Size (Kg)
A	1	2200	1750	8000
B	3	1600	1500	8000
C	3	1500	1000	8000
D	0	1500	1850	8000
E	2	1750	1900	8000
F	3	2400	1500	8000
G	2	2300	2000	4000
H	1	2100	2000	4000
I	4	2500	2750	4000
J	2	1250	1800	4000
K	3	1500	2300	4000
L	1	2300	2250	4000
M	2	1750	1800	4000

e. Processing changeover times (in hours)

	A	B	C	D	E	F	G	H	I	J	K	L	M
A	0	1	1	1	1	1	1	1	1	1	1	1	1
B	1	0	1	1	1	1	1	1	1	1	1	1	1
C	1	1	0	1	1	1	1	1	1	1	1	1	1
D	1	1	1	0	1	1	1	1	1	1	1	1	1
E	1	1	1	1	0	1	1	1	1	1	1	1	1
F	1	1	1	1	1	0	1	1	1	1	1	1	1
G	1	1	1	1	1	1	0	0	0	0	0	0	0
H	1	1	1	1	1	1	0	0	0	0	0	0	0
I	1	1	1	1	1	1	0	0	0	0	0	0	0
J	1	1	1	1	1	1	0	0	0	0	0	0	0
K	1	1	1	1	1	1	0	0	0	0	0	0	0
L	1	1	1	1	1	1	0	0	0	0	0	0	0
M	1	1	1	1	1	1	0	0	0	0	0	0	0

f. Packaging changeover times (in hours)

	A	B	C	D	E	F	G	H	I	J	K	L	M
A	0	2	2	2	2	2							
B	1	0	2	2	2	2							
C	1	1	0	2	2	2							
D	1	1	1	0	2	2							
E	1	1	1	1	0	2							
F	1	1	1	1	1	0							
G							0	2	2	2	2	2	2
H							1	0	2	2	2	2	2
I							1	1	0	2	2	2	2
J							1	1	1	0	2	2	2
K							1	1	1	1	0	2	2
L							1	1	1	1	1	0	2
M							1	1	1	1	1	1	0

Appendix 3: Nomenclature

Sets

$i \in$ Product Mix;

$b \in$ Batch of a Product Mix;

$u \in$ Pasteurizer;

$v \in$ Vessel;

$w \in$ Packaging lines;

Subsets

$PasteurizersFor_i$	Pasteurizers for processing Product Mix *i*;
$VesselsFor_i$	Vessels for processing Product Mix *i*;
$FreezersFor_i$	Freezers for processing Product Mix *i*;
$PackingLinesFor_i$	Packaging lines for processing Product Mix *i*;

Parameters

$Fillingrate_u$	Pasteurization rate for Pasteurizer *u*;
$AgingTime_i$	Aging time for Product Mix *i;*
$FreezerTime_i$	Freezer time for Product Mix *i*;
$Emptyrate_i$	Packaging rate for Product Mix *i*;
$MinVesselSize_i$	Minimum vessel size for Product Mix *i*;
$Demand_i$	Demand for Product Mix *i*;
$ProcessChOTimes_{ii'}$	Process change-over time between Product Mixes *i* and *i'*;
$PackageChOTimes_{ii'}$	Packaging line change-over time between Product Mixes *i* and *i'*;
n	Maximum number of weeks (= 7);
N	Set of processing week (1 to *n*);

$Week$ Number of working hours per weeks (= 120 hours);

$idle$ Changeover time to idle state (= 2 hours);

Parameter functions

Minimum number of batches for Product Mix *i*; $MinNoBatch_i = {Demand_i}/{MinVesselSize_i}$

Pasteurization time for Product Mix *i* ; $FillingTime_i = {MinVesselSize_i}/{Fillingrate_i}$

Packaging time for Product Mix *i*; $EmptyTime_i = {MinVesselSize_i}/{EmptyTime_i}$

Working hours:1 when the facility is open & 0 when the facility is closed for weekend breaks;

$$stepFunction\ WeekendBreak = \left\{\begin{array}{l} 1\ if\ 0 < t \le (l*Week) - 2 \\ 0\ if\ (l*Week) - 2 < t \le l*Week \end{array}\right\}\ \forall t,\ t \in 1..n*Week,\ l \in N$$

Interval Variables (Decision Variable)

$FillProcess_{ibu}$ Pasteurization interval for Product Mix *i*, batch *b* in unit *u* (optional);

$FillAssign_{ib}$ Pasteurization interval for assigning Product Mix *i*, and batch *b*;

$FreezeProcess_{ibx}$ Freezing interval for Product Mix *i*, and batch *b* in freezer *x* (optional);

$FreezeAssign_{ib}$ Freezing interval for assigning Product Mix *i*, and batch *b*;

$EmptyProcess_{ibw}$ Packaging interval for Product Mix *i*, batch *b* in line *w* (optional);

$EmptyAssign_{ib}$ Packaging interval for assigning Product Mix *i*, and batch *b*;

$VesselProcess_{ibv}$ Vessel interval for aging Product Mix *i*, batch *b* in vessel *v* (optional);

$VesselAssign_{ib}$ Vessel interval for assigning Product Mix *i*, and batch *b*;

$WaitProcessV_{ib}$ Waiting interval for assigning Product Mix *i*, and batch;

$AgeProcessV_{ib}$ Aging interval for assigning Product Mix *i*, and batch *b*;

Sequence Variables (Decision Variable)

$PasturSeq_u$ Pasteurization sequence for intervals $FillProcess_{ibu}$ in pasteurizer *u*;

$VesselSeq_v$ Vessel sequence for intervals $VesselProcess_{ibv}$ in vessel *v*;

$PackSeq_w$ Packaging sequence for intervals $EmptyProcess_{ibw}$ in line *w*;

$FreezerSeq_x$ Freezing sequence for intervals $FreezeProcess_{ibx}$ in freezer *x*;

ILOG IBM CPLEX built-in modeling variables and functions (IBM, 2016)

Interval decision variable:- Variable time interval whose exact position is yet to be determined.

Sequence decision variable:- Variable to determine the order of interval decision variables.

stepFunction:- a function to create a step-wise function (to create varying value 0-slope graphs)

noOverlap:- constraint function to prevent interval variables overlapping in a sequence.

forbidExtent:- constraint function to prevent interval variables overlapping a given time period.

alternative:- constraint function for creating an encapsulating interval over low-level intervals

startAtStart:- precedence constraint linking the start of a given interval with the start of another

endAtEnd:- precedence constraint linking the end of a given interval with the end of another

startAtEnd:- precedence constraint linking the start of a given interval with the end of another

endBeforeStart:- precedence constraint to arrange the start of a given interval after the end of another

startOf:- variable that gives the starting time of a given interval variable

endOf:- variable that gives the ending time of a given interval variable

lengthOf:- variable that gives the length of time for a given interval variable (*endOf* – *startOf*)