ORIGINAL PAPER

# A branch-and-price algorithm for parallel machine campaign planning under sequence dependent family setups and co-production

Serkan Kalay and Z. Caner Taşkın

Department of Industrial Engineering, Boğaziçi University, 34342 Bebek, İstanbul, Turkey

**ARTICLE HISTORY**
Compiled June 12, 2021

**ABSTRACT**
We investigate a tactical level production planning problem in process industries under costly sequence dependent family setups, which drives the need for manufacturing of product families in campaigns. The motivation of our work is float glass manufacturing, which has some unique properties such as co-production and uninterruptible production. To solve the problem, we develop a branch-and-price algorithm based on a novel formulation of the problem. Our algorithm is capable of providing consistent performance in different problem sizes. We compare our proposed algorithm with previous work on the same problem by running computational experiments.

**KEYWORDS**
Branch-and-price; column generation; sequence dependent family setup; MILP; process industry

## 1. Introduction

In this paper, we study a production planning problem in process industries in the presence of sequence dependent family setups and co-production on parallel machines. Since process industries are usually capital intensive, cost effectiveness is critical within the manufacturing process and its planning.

Manufacturing, transportation, inventory holding and demand satisfaction related costs are often directly considered in supply chain planning. Nevertheless, loss of efficiency in production line capacity usage can have significant impact on the overall effectiveness especially in process industries. Setup times and, if the nature of the process imposes, co-production need to be dealt with to improve effectiveness.

The motivation of our work stems from float glass manufacturing, which is an example of process manufacturing such that the cost is the main driver in major decisions in planning process. This brings in complexities in production planning in addition to special characteristics of the process itself.

The term float refers to the physical nature of the glass production. Molten solution, which consists of raw materials such as sand, limestone and soda ash, fed into a bath and obtains its flat shape by floating over tin bath due to the

CONTACT Serkan Kalay. Email: serkan.kalay@boun.edu.tr

difference in density. The furnace melting the mixture operates on 24/7 basis. The solution cools down and becomes solid in the annealing step. Finally, the glass sheet is cut into different sizes before being picked up and stacked in storage area. Figure 1 illustrates the float glass production.

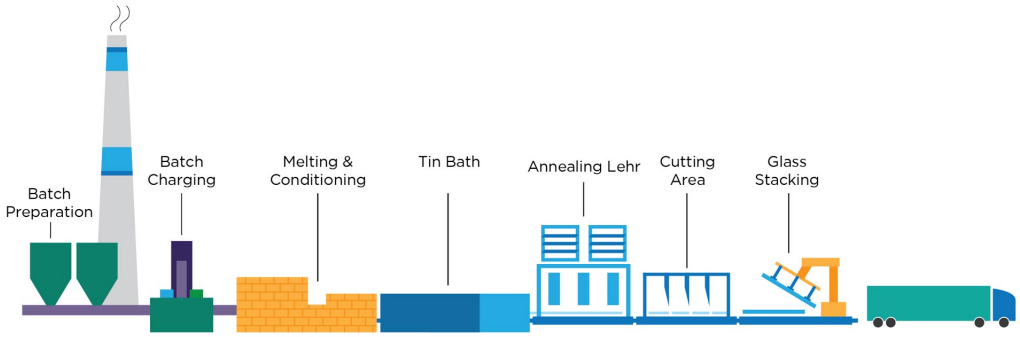


**Figure 1.** An illustrative float glass furnace and production line (Glass (2019))

Random errors on the glass surface appear during production due to the chemical nature of the process. There can be different types of error such as visually detectable defects. There can also be serious errors which would enforce the output to be scrapped (Taşkın and Ünal (2009)). Depending on where the final product is used, some of these errors can be disregarded. Hence, we can categorize glass with different errors into quality groups. Depending on the cutting decisions regarding the size of the final product, the line can yield different size group and quality combinations. Using historical data, planners can determine the maximum yield percentages for product groups such that the products have the same size group and quality. This results in producing several different products together by necessity with a limited control through cutting decisions, which is defined as partially controllable co-production. We refer to Taşkın and Ünal (2009) for further information on co-production in float glass manufacturing.

The color of the glass is controlled by chemical additives to the solution, and switching from one color to another requires all the glass produced to be broken from the beginning of switch until the desired color is actually obtained (Taşkın and Ünal (2009)). Since the furnace consumes high amount of energy, setup decisions are critical in terms of cost effectiveness. Moreover, switching from, for instance, blue glass to clear does not take as much time as switching from blue to green, which implies the sequence dependency. In addition to color, inline coating of the glass is another attribute that requires setup. We note here that float glass manufacturing is a continuous process and at any given time there is a color that is being actively produced. Hence, we ignore the starting setups, which happens only after the demolishment of the furnace and its reconstruction.

Expensive nature of the setups require products to be produced in batches at least for a certain amount of time, in order to compensate the associated costs. Hence, the products are grouped into families with respect to their attributes that determine the setups. Production run of a product family is called a cam-

paign, which requires to be efficiently planned in terms of timing and duration as it has impact on demand satisfaction and inventory projection. Campaign planning is the process of determining the timing and the length of production run decisions for families (Kalay and Taşkın (2020)).

Campaign planning is typically executed on a monthly basis since demand forecasts become available in monthly resolution (Kalay and Taşkın (2020)). We refer to Figure 2 presented by Kalay and Taşkın (2020) for the illustration of the main components of the campaign planning problem. Demand forecast being available in discrete time requires the demand satisfaction plan also to be in discrete time, whereas other input data such as production rates, setup durations are expressed in continuous time. A method to solve campaign planning problem needs to incorporate this incompatibility while generating a cost effective plan. Moreover, the output campaign plan itself needs also to be in continuous time, which makes the problem differ from aggregate planning.

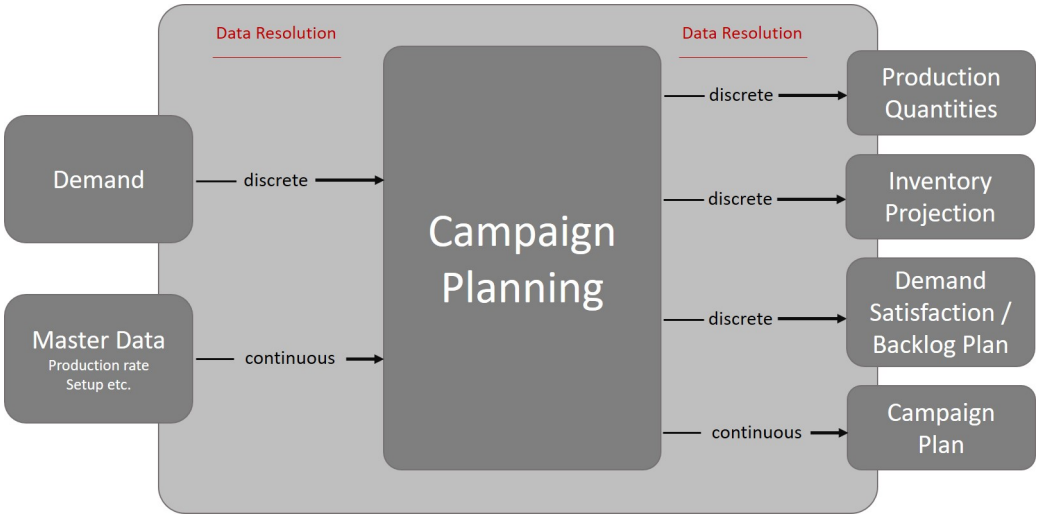


**Figure 2.** Main inputs and outputs of Campaign Planning problem (Kalay and Taşkın (2020))

Let us illustrate the campaign planning problem with a small numerical example. Let $R1$ and $R2$ be two parallel machines and $P1$, $P2$, $P3$, $P4$ be four products with $P1$ and $P2$ are clear glass denoted as color RZ whereas $P3$ and $P4$ are green glass denoted as color YS. Table 1 shows details for products. Table 2 shows the production alternatives and Table 3 shows the setup information for colors on each resource. For the sake of simplicity, we only include color as family and consequently setup factor as well as co-production phenomenon. All the products have demands for 3 periods as shown in Table 4. Figure 3 shows the optimal campaign plan for the numerical example.

| Product | Color | Inventory Holding Cost | Backlog Cost |
|---|---|---|---|
| P1 | RZ | 6.00 | 360.00 |
| P2 | RZ | 2.80 | 210.00 |
| P3 | YS | 40.40 | 570.00 |
| P4 | YS | 49.20 | 1050.00 |

**Table 1.** Products in illustrative example

| Product | Resource | Daily Rate (tons) | Cost (per ton) |
|---|---|---|---|
| P1 | R1 | 192.86 | 100 |
| P1 | R2 | 153.46 | 150 |
| P2 | R1 | 153.46 | 50 |
| P2 | R2 | 207.69 | 90 |
| P3 | R1 | 207.69 | 360 |
| P3 | R2 | 168.75 | 325 |
| P4 | R1 | 160.59 | 243 |
| P4 | R2 | 153.46 | 452 |

**Table 2.** Production alternatives for illustrative example

| Resource | From Color | To Color | Duration | Cost |
|---|---|---|---|---|
| R1 | RZ | YS | 2.3 | 32100 |
| R1 | YS | RZ | 5 | 94600 |
| R2 | RZ | YS | 5 | 55700 |
| R2 | YS | RZ | 9.6 | 175000 |

**Table 3.** Color setups for illustrative example

| Product | Period | Quantity |
|---|---|---|
| P1 | 1 | 10000 |
| P1 | 2 | 15000 |
| P1 | 3 | 5000 |
| P2 | 1 | 7500 |
| P2 | 2 | 9000 |
| P2 | 3 | 7500 |
| P3 | 1 | 1000 |
| P3 | 2 | 6500 |
| P3 | 3 | 500 |
| P4 | 1 | 6000 |
| P4 | 2 | 8500 |
| P4 | 3 | 750 |

**Table 4.** Product demands for illustrative example

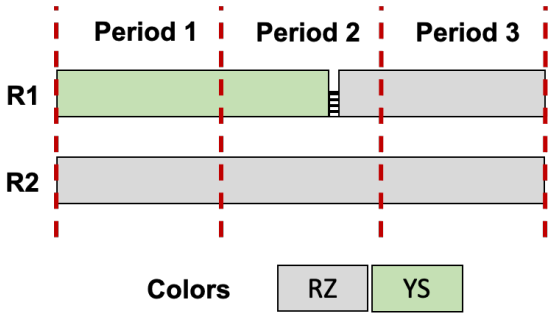


**Figure 3.** Optimal campaign plan for illustrative example.

| | Period 1 | | | Period 2 | | | Period 3 | | |
|---|---|---|---|---|---|---|---|---|---|
| | **R1** | **R2** | **End Inv.** | **R1** | **R2** | **End Inv.** | **R1** | **R2** | **End Inv.** |
| **P1** | 0 | 10000 | 0 | 0 | 15000 | 0 | 5000 | 0 | 0 |
| **P2** | 0 | 5507 | -1993 | 3378 | 564 | -7051 | 11169 | 12896 | 9514 |
| **P3** | 8000 | 0 | 7000 | 0 | 0 | 500 | 0 | 0 | 0 |
| **P4** | 6104 | 0 | 104 | 9146 | 0 | 750 | 0 | 0 | 0 |

**Table 5.** Solution to illustrative example.

We observe that $R1$ is dedicated to producing $YS$ for the first period and part of the second period. Then it is planned to switch to produce clear glass. $R2$ is producing clear glass for all 3 periods. Finally, Table 5 shows the solution in detail. We read the amount produced on each machine for each period and projected ending inventory for the corresponding period. Please note that a negative value in column **End Inv.** denotes backlog for the product.

Kalay and Taşkın (2020) propose efficient mixed-integer linear programming formulations of the campaign planning problem on a single machine. Our study extends this study to parallel machines. Since the parallel machine case is computationally more challenging than the single machine case, we also propose a branch-and-price algorithm in this paper. Moreover, we keep the co-production phenomenon within the scope of our study since co-production is a crucial aspect of float glass manufacturing. We note that despite our focus on float glass manufacturing, this study can be generalized to other process industries such as oil refineries, various chemical processes of pharmaceuticals and food and beverage. Extension to oil industries can particularly be beneficial since they also have co-production phenomenon.

The remainder of the paper is organized as follows. Section 2 gives an overview of the related work. Section 3 explains the mathematical formulations. We explain details of our branch-and-price (B&P) algorithm in Section 4. Section 5 presents the results of computational experiments and finally Section 6 concludes the paper.

## 2. Literature Review

Process industries are capital oriented and as a result the main driver within the manufacturing process is the cost effectiveness. In float glass manufacturing, raw materials including sand, soda ash, limestone etc. are molten in a furnace, which needs to operate on 24/7 basis. The energy consumption of the furnace is cost intensive, hence needs to be properly managed in production planning. Taşkın and Ünal (2009) explain the unique properties of the production environment such as random yields, partially controllable co-production and product substitution, and propose a MIP formulation to efficiently plan the production based on pre-defined color campaigns. The model determines the campaign durations and product composition to meet customer demand. Miegeville (2005) studies the float glass manufacturing process and also develops a MIP for production planning. Their model determines whether a specific product is produced in a time period. The model, however, does not address the sequence dependent family setups, which is complex being dependent on both color and coating attributes. Kalay and Taşkın (2020) also study the float glass manufacturing problem on a single machine and they propose two MIP formulations to address both the campaign duration and sequencing, hence the setups.

Co-production is the phenomenon of producing a family of several different products simultaneously, which occurs usually due to physical or chemical properties of the system. To embody the formal definition of co-production, we refer to Ağralı (2012). The author studies uncapacitated lot sizing problem and shows that it can be reduced to single item lot sizing problem to be solved by dynamic programming. Rafiei, Nourelfath, Gaudreault, Santa-Eulalia, and Bouchard (2015) formulate

a MIP for exactly four product families in wood remanufacturing industry with co-production. They propose a re-planning based two phase solution procedure and test it via simulation. The backorder performance improves with proposed approach. Gaudreault, Frayret, Rousseau, and D'Amours (2011) study planning and scheduling in lumber industry, which contains co-production and alternative process selection. They compare MIP and CP formulations and show that MIP is unstable in terms of performance between different datasets whereas CP provides good quality solutions in shorter time. Complex nature of the planning problem in refineries results in scientific work concentration on subsytems Joly, Moro, and Pinto (2002). The authors formulate a MINLP containing two main parts. The model employ the operating rules of the processes and flow constraints. Moreover, viscosity contraints cause co-production for which types and rates of co-products are known in advance.

Susarla and Karimi (2011) study campaign planning in batch plants on parallel machines with alternative selection. Demand due dates are associated with macro periods while campaign allocations are tracked on micro periods. Macro periods correspond to time between consecutive demands, whereas micro periods correspond to campaign slots within a macro period. Their formulation allows spanning multiple micro periods for only the last campaign in the horizon. Macro periods generally correspond to higher level decisions whereas micro periods correspond to operational or more detailed decisions which can also translate into recourse actions. For instance, a month can be defined as a macro period and days of the month can be micro periods associated. Camargo, Toledo, and Almada-Lobo (2012) propose three formulations, which differ from each other with respect to the time representation. In discrete representation, there exists both macro and micro periods, whereas in continuous representation the formulation determines the sequence and the duration of available common resource batches. Finally, they propose a hybrid representation which is based on batch scheduling within periods. Their results favor the discrete formulation since it is more compact compared to others and is able to provide good solutions. Kopanos, Puigjaner, and Maravelias (2011) study the problem in the existence of sequence dependent family setups. Their formulation allows setup crossover between periods. They also extend their original formulation with a dummy product which allows to model continuous production. Almada-Lobo, Klabjan, Maria, Carravilla, and Oliveira (2010) study the production planning in glass container manufacturing. There needs to be an integer number of mold cavities planned, which makes the problem hard and the authers apply Lagrangian decomposition based on these variables. They aim to improve the efficiency of the plan by penalizing the production losses. To improve the lower bounds the authors add valid inequalities the impact of which increases for more complex run instances. Ghirardi and Ameiro (2019) study lot sizing and scheduling on unrelated parallel machines in the existence of backordering and setup carryover. They develop three matheuristics driven by local search, local branching and feasibility pump. Their experiments show that local search outperforms other algorithms and base formulation solved by two MIP solvers separately. Guimarães, Klabjan, and Almada-Lobo (2014) tackle lot sizing and scheduling problem with sequence dependent setup times. They propose two formulations for the problem. One decides specifically on setup between products whereas the second model allocates a sequence of products to a machine for a period. Their work does not explicitly mention how to come up with pre-defined sequences. Furthermore,

their model does not allow for setup crossover, which is necessary in settings where setup durations are incompatible with micro-period lengths of the model.

MIP based formulations are also applied in the existence of pre-defined jobs, namely scheduling and sequencing based problems. Vieira, Pinto-Varela, Moniz, Barbosa-Póvoa, and Papageorgiou (2016) use a MIP based approach to schedule pre-assigned lots in biopharmaceutical processes. The planning horizon is divided into periods, the length of which is determined through a decision variable. Shelf-life is also considered in the formulation via a variable counting storage time. Kramer, Iori, and Lacomme (2019) propose five novel MIP formulations for identical parallel machine scheduling with family dependent setups. The formulations are inspired by single commodity, arc-flow and set covering formulations. They conduct extensive set of numerical experiments and show the efficiency of two of the formulations driven from strong bounds. Hinder and Mason (2017) study the same problem on a single machine. They formulate a model exploiting properties of optimal solutions. They argue that the LP relaxation of their formulation is stronger than other formulations in the literature. Hence, the model is able to find optimal solutions for instances with relatively high number of families and long setup times.

Chen and Powell (1999) develop a general framework based on Dantzig-Wolfe decomposition applicable to parallel machine scheduling problem with an objective consisting of additive criterion. The authors suggest a set-partitioning reformulation, which leads to design of exact branch-and-price algorithms. The computational experiments on minimizing weighted number of tardy jobs and total weighted completion time shows the method being promising for solving large problems. However, the method requires sophisticated integer program and set partitioning problem, which may result in extra work modifying the original problem. A branch-and-cut-and-price algorithm is presented in Pessoa, Uchoa, Poggi, and De Freitas Rodrigues (2010) for identical parallel machines. The algorithm employs dynamic programming to fix variables. Numerical experiments show the approach can solve medium sized problems to optimality but requires further improvements to be efficient for larger problems. Moreover, the method can also be used for solving in a class of vehicle routing problems. Akker, Hoogeveen, and Kempen (2012) study the same problem with minmax objective functions. Additionally jobs have release dates and precedence constraints. Authors apply column generation to obtain lower bound and try to determine a feasible schedule by solving an integer linear program. Numerical results on tested instances show that the derived lower bounds are equal to optimal value. The approach, however is prone to performance decrease with changes to the problem such as unrelated machines. Yin, Chen, Qin, and Wang (2018) propose a two agent scheduling problem, where agents have competing objectives. The method proposed is a branch-and-price algorithm where the pricing problem reduces to a single machine scheduling problem solved efficiently by dynamic programming for tested instances. However, they note that for larger instances a pseudo-polynomial algorithm may not be as efficient. The branching rule is based on the original variables, which translates into restricting positions for a subset of jobs and does not have a significant impact on the pricing problem solution efficiency. The method is able to provide good quality bounds and solves to optimality within a small number of iterations. Xiong, Zhou, Yin, Cheng, and Li (2019) develop a branch-and-price method to solve multitasking scheduling problem on unrelated parallel machines. Authors propose a greedy algorithm that exploits

the characteristics of optimal schedule to generate initial columns. The algorithm uses a novel approach combining genetic algorithm and dynamic programming to efficiently determine the promising columns and employs in-out column generation, which utilizes a specialized approach for reduced cost calculation based on the separation point defined as convex combination of an interior and an exterior dual solution space. The main aim of this approach is to keep the dual variables stable. Numerical studies provide insights on the efficiency of multitasking to the decision maker.

Seeanner, Almada-Lobo, and Meyr (2013) study the multi-level lot sizing and scheduling problem with sequence dependent setups and they formulate a MIP that allows setup changeover over period boundaries. Since the problem is hard to solve to optimality, they apply product, resource and process based decompositions applied sequentially and develop an improvement heuristic based on variable neighborhood search (VNS) and fix-and-optimize. James and Almada-Lobo (2011) propose an iterative MIP based decomposition heuristic for single and parallel machine lot sizing and scheduling problem. The improvement heuristic is stochastic as opposed to a deterministic relax-and-fix algorithm. For the initial solution generation, they use a rolling-horizon based construction heuristic. Camargo, Toledo, and Almada-Lobo (2014) develop a new exact algorithm for the lot sizing and scheduling problem in spinning industry. The algorithm combines the solutions generated by B&B with a problem specific procedure, which injects new and better upper bounds into the original problem. Farahani, Grunow, and Günther (2012) extend the production planning with distribution plan for perishable food production. The authors employ a hierarchical modelling approach with order batching, production planning and distribution subproblem. Since the problem also contains the need for minimizing the quality decay of products, the solution provided by proposed approach can guarantee a certain quality without improving the costs much. Bektur and Saraç (2019) study unrelated parallel machine scheduling problem with sequence dependent setup times and machine eligibility restrictions. They compare a simulated annealing (SA) algorithm against a tabu search(TS). The latter is shown to yield better solutions with long-term memory. In the presence of pre-defined lots Schaller (2014) compare three metaheuristics for scheduling identical parallel machines minimizing total tardiness with sequence dependent family setups. According to their results, genetic algorithm outperforms tabu search and optimal B&B.

Capacitated Lot Sizing Problem (CLP) is the fundamental production planning problem and is known to be NP-hard Florian, K. Lenstra, and H. G. Rinnooy Kan (1980). In this study, we focus on parallel machine case, which can be classified as General Lot Sizing Problem (GLSP) with sequence dependent family setup and co-production extensions. Finding a feasible solution for single-level special case of General Lot Sizing Problem for Multiple Production Stages (GLSPMS) is NP-complete Fleischmann and Meyr (1997). Kalay and Taşkın (2020) propose MIP formulations that incorporate continuous and discrete input data for lot sizing and scheduling problem with sequence dependent family setups on a single machine. We generalize our earlier work to parallel unrelated machines. Since a straightforward extension of our earlier models are unable to solve parallel machine instances efficiently, in this paper we introduce a MILP formulation with a novel approach in representing the entire planning horizon resulting in a compact model and an efficient B&P algorithm. We also note that our approach contributes to the existing branch-and-price based studies in the

sense that the problem includes not only pre-defined jobs but also lot sizing.

## 3. Mathematical models

### *3.1. Family Transition Based Model Variant on Parallel Machines*

In this section we extend the mathematical formulation proposed by Kalay and Taşkın (2020) to solve the campaign planning problem in process industries on a single machine to parallel machines. The authors formulate two mixed integer linear models and their variants. All of the models are based on allocating a *pattern* to the machine for each production period.

Products are grouped into families with respect to their color and coating attributes, which determine the sequence dependent setups. A pattern is an ordered list of families that will be produced consecutively within a period while respecting the minimum production duration of each family. Consequently, assigning a pattern to a period will enable production of products of families that exist within the selected pattern. Optimization models proposed in Kalay and Taşkın (2020) calculate optimal production quantities of products in alignment with pattern assignments. Figure 4 illustrates pattern examples. Kalay and Taşkın (2020) explain that the patterns respect the setup feasibility and times of families appearing within a pattern. For instance, setup should be feasible between families FM and MV in order Pattern 2 to be feasible. Similarly, from family BR to MV and from MV to FM for Pattern 3 and from MV to BR as well as BR to MV for Pattern 4.

We adapt the pattern generation and pre-processing algorithms proposed by Kalay and Taşkın (2020) to build set of patterns. Their algorithms work for a single machine. The generation algorithm takes the set of families and corresponding setup matrix as input and recursively builds patterns by checking whether a family can be added to the pattern considering the minimum production and setup durations as well as period length. The patterns generated in this way are a subset of all permutations of families. There are three main reasons for not generating all permutations. For some family pairs, the setup may not be feasible, hence need not be generated. Second, for patterns which contain the same distinct set of families and same starting and ending family, the pattern with least cost will be sufficient to be included in set of patterns, since more costly patterns will not preferable in an optimal solution. Third, setup and minimum production durations of selected families must not exceed period length.

We run the algorithm for each machine separately for parallel machines instances. Their algorithm is pseudo-polynomial with computational complexity being expressed as $O(\lceil |T|/(min|s+f|)\rceil n)$ where $n$ is number of families, $T$ is length of a period and $min|s+f|$ being the minimum possible duration of setup duration for family $f$ and the corresponding minimum production duration. The algorithm will try to extend an existing pattern while ensuring that the minimum setup and minimum production duration can fit within $T$. For parallel machine instance, the complexity becomes $O(\lceil |T|/(min|s+f|)\rceil nr)$ since we run the algorithm for each machine $r$. We note that the algorithms are efficient in practice since the minimum production and setup durations do not allow for large number of families being added to a pattern, which reduces the running

time. Moreover, they also keep track of the amount produced for a family at the beginning, in the middle or at the end of a period so that they can properly manage the minimum production durations and setups between adjacent periods.

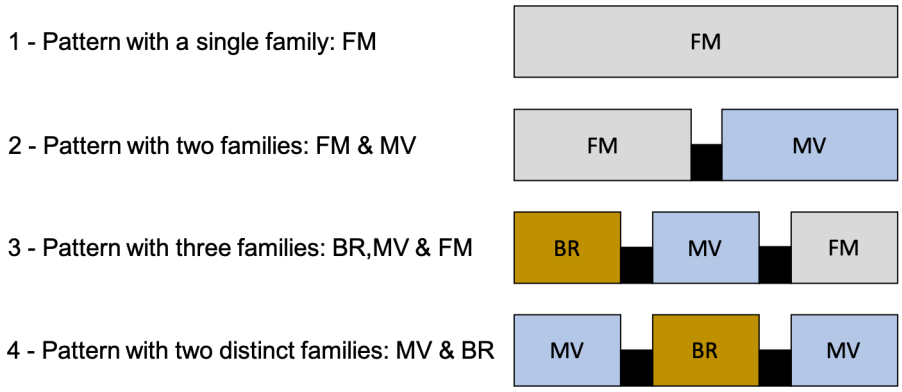


**Figure 4.** Sample illustrations for patterns including up to 3 families

| **Set** | **Description** |
|---|---|
| $J$ | Set of products |
| $R$ | Set of production lines |
| $Q$ | Set of quality groups |
| $S$ | Set of size groups |
| $T$ | Set of time periods |
| $P$ | Set of campaign patterns |
| $F$ | Set of product families |
| $O$ | Set of orders for timing of production in a period (b: beginning, m: middle, e: end) |
| $P(f)$ | Set of patterns containing family $f$ |
| $F(p)$ | Set of families belonging to pattern $p$ |
| $F^o(p)$ | Set of families appearing in order $o$ in pattern $p$ |
| $J(f)$ | Set of products belonging to family $f$ |
| $\Gamma^r(f,g)$ | Set of product family couples that are infeasible on production line $r$, $f,g \in F$ (i.e. production of family $g$ is not possible to immediately start after family $f$) |
| $P^S(f)$ | Patterns that family $f$ is the first family |
| $P^E(f)$ | Patterns that family $f$ is the last family |
| **Parameter** | **Description** |
| $D_{jt}$ | Demand of product $j$ in period $t$ |
| $I_{j(-1)}$ | Beginning inventory of product $j$ |
| $v_{jr}$ | Production speed of product $j$ on production line $r$ |
| $A_t$ | Available capacity of production lines in period $t$ |
| $S(j)$ | Index of the size group of product $j$ |
| $Q(j)$ | Index of the quality group of product $j$ |
| $R_{fqsr}$ | Maximum production ratio for quality group $q$ and size group $s$ for family $f$ on production line $r$ |
| $MD_{fr}$ | Minimum production duration for family $f$ on production line $r$ |
| $NT_{fp}$ | Number of times family $f$ appears in the middle order of pattern $p$ |
| $ST_{pr}$ | Setup time needed for family order within pattern $p$ on production line $r$ |
| $ST_{fgr}$ | Setup time needed for switching from product family $f$ to family $g$ on production line $r$ |
| $h_j$ | Inventory holding cost for product $j$ |
| $b_j$ | Cost of backlogging a demand of product $j$ for a single period |
| $u_{jr}$ | Unit production cost for producing product $j$ on line $r$ |
| $c_{fgr}$ | Setup cost for switching from family $f$ to family $g$ on production line $r$ |
| $c_{pr}$ | Total setup cost for family order within pattern $p$ on production line $r$ |
| **Variable** | **Description** |
| $I_{jt}$ | Inventory of product $j$ at the end of period $t$ |
| $S_{jtk}$ | Satisfied quantity of demand from period $t$ of product $j$ in period $k$ |
| $U_{jt}$ | Unsatisfied quantity of demand from period $t$ of product $j$ |
| $X_{jrt}$ | Production quantity of product $j$ on production line $r$ in period $t$ |
| $d^o_{frt}$ | Number of days spent for production of family $f$ in order $o$ on production line $r$ in period $t$ |
| $\delta_{prt}$ | Binary indicator variable for selection of pattern $p$ on production line $r$ in period $t$ |
| $\gamma^S_{frt}$ | Indicator for selection of family $f$ as starting on production line $r$ in period $t$ |
| $\gamma^E_{frt}$ | Indicator for selection of family $f$ as ending on production line $r$ in period $t$ |
| $\theta_{fgrt}$ | Auxiliary variable indicating whether production line $r$ switched from family $f$ to family $g$ at the beginning of period $t$ |
| $n^P_{fgrt}$ | Number of days spent for setup on production line $r$ in predecessor period $t-1$ for switched from family $f$ to family $g$ at the beginning of period $t$ |
| $n^S_{fgrt}$ | Number of days spent for setup on production line $r$ in successor period $t$ for switched from family $f$ to family $g$ at the beginning of period $t$ |
| $F_{rt}$ | Setup time spent on production line $r$ at the beginning of period $t$ |
| $B_{rt}$ | Setup time spent on production line $r$ at the end of period $t$ |

**Table 6.** Symbols used in FTBMV-PM

We extend the Family Transition Based Model Variant (FTBMV) of the four models presented, which is shown to outperform other three. We use set of patterns $P$ generated by algorithms adapted to parallel machine instances. We call this new model is Family Transition Based Model on Parallel Machines (FTBMV-PM). Table 6 lists the symbols used in FTBMV-PM along with their brief descriptions.

**Model 1.** *Family Transition Based Model Variant on Parallel Machines (FTBMV-PM)*

$$
\begin{aligned}
min \quad & \sum_{j\in J}\sum_{t\in T}\left[h_j\ I_{jt} + b_j\ (|T|-t+1)\ U_{jt} + \sum_{\substack{k\in T\\ k\le t}}(b_j\ (t-k)\ S_{jkt})\right] \\
& + \sum_{r\in R}\sum_{j\in J}\sum_{t\in T} u_{jr}X_{jrt} + \sum_{r\in R}\sum_{p\in P} c_{pr}\delta_{pr} \\
& + \sum_{r\in R}\sum_{t\in T}\sum_{(f,g)\notin\Gamma^r(f,g)} c_{fgr}\theta_{fgrt}
\end{aligned}
$$

*subject to*

$$\sum_{\substack{k\in T\\ k\ge t}} S_{jtk} + U_{jt} = D_{jt} \qquad \forall\ j\in J,\ t\in T \tag{1}$$

$$I_{j(t-1)} + \sum_{r\in R} X_{jrt} - \sum_{\substack{k\in T\\ k\le t}} S_{jkt} = I_{jt} \qquad \forall\ j\in J,\ t\in T \tag{2}$$

$$\sum_{j\in J} v_{jr}X_{jrt} = \sum_{o\in O} d^o_{frt} \qquad \forall\ f\in F,\ r\in R,\ t\in T \tag{3}$$

$$\sum_{\substack{j\in J(f)\\ Q(j)\le q\\ S(j)\le s}} X_{jrt} \le \sum_{j\in J(f)} X_{jrt}\ R_{fqrs} \qquad \forall\ r\in R,\ f\in F,\ q\in Q,\ s\in S,\ t\in T \tag{4}$$

$$\sum_{p\in P}\delta_{prt} = 1 \qquad \forall\ r\in R,\ t\in T \tag{5}$$

$$d^m_{frt} \ge MD_{fr}NT_{fp}\delta_{pt} \qquad \forall\ r\in R,\ p\in P,\ f\in F^m(p),\ t\in T \tag{6}$$

$$d^e_{frt} + d^b_{fr(t+1)} \ge MD_{fr}\delta_{prt} \qquad \forall\ r\in R,\ p\in P,\ f\in F^b(p)\cup F^e(p),\ t\in T \tag{7}$$

$$d^o_{frt} \le \sum_{p\in P^o(f)} A_t\delta_{prt} \qquad \forall\ r\in R,\ f\in F,\ o\in O,\ t\in T \tag{8}$$

$$\sum_{f\in F}\sum_{o\in O} d^o_{frt} + \sum_{p\in P} ST_{pr}\delta_{prt} + F_{rt} + B_{rt} = A_t \qquad \forall\ r\in R,\ t\in T \tag{9}$$

$$\gamma^S_{frt} = \sum_{p\in P^S(f)} \delta_{prt} \qquad \forall\ r\in R,\ f\in F,\ t\in T \tag{10}$$

$$\gamma^E_{frt} = \sum_{p\in P^E(f)} \delta_{prt} \qquad \forall\ r\in R,\ f\in F,\ t\in T \tag{11}$$

$$\theta_{fgrt} \le \gamma^E_{fr(t-1)} \qquad \forall\ r\in R,\ f,g\in F,\ t\in T,\ t\ge 1 \tag{12}$$

$$\theta_{fgrt} \le \gamma^S_{grt} \qquad \forall\ r\in R,\ f,g\in F,\ t\in T \tag{13}$$

$$\theta_{fgrt} \ge \gamma^E_{fr(t-1)} + \gamma^S_{grt} - 1 \qquad \forall\ r\in R,\ f,g\in F,\ t\in T,\ t\ge 1 \tag{14}$$

$$n^P_{fgrt} + n^S_{fgrt} = ST_{fgr}\theta_{fgrt} \qquad \forall\ r\in R,\ f,g\in F,\ (f,g)\notin\Gamma^r(f,g),\ t\in T \tag{15}$$

$$F_{rt} = \sum_{(f,g)\notin\Gamma^r(f,g)} n^S_{fgrt} \qquad \forall\ r\in R,\ f,g\in F,\ t\in T \tag{16}$$

$$B_{rt} = \sum_{(f,g)\notin\Gamma^r(f,g)} n^P_{fgr(t+1)} \qquad \forall\ r\in R,\ f,g\in F,\ t\in T \tag{17}$$

$$\sum_{\substack{f,g \in F \\ (f,g) \notin \Gamma^r(f,g)}} \theta_{fgrt} = 1 \qquad \forall\, r \in R,\ t \in T,\ t \geq 1 \tag{18}$$

$$I_{jt}, U_{jt} \geq 0 \qquad \forall (j,t) \tag{19}$$

$$S_{jtk} \geq 0 \qquad \forall (j,t,k \geq t) \tag{20}$$

$$d^o_{frt} \geq 0 \qquad \forall f,r,t \tag{21}$$

$$X_{jrt} \geq 0 \qquad \forall (j,r,t) \tag{22}$$

$$\delta_{prt} \in \{0,1\} \qquad \forall (p,r,t) \tag{23}$$

$$0 \leq \theta_{fgrt} \leq 1 \qquad \forall (f,g,r,t) \tag{24}$$

$$\gamma^S_{frt}, \gamma^E_{frt} \geq 0 \qquad \forall (f,r,t) \tag{25}$$

$$F_{rt}, B_{rt} \geq 0 \qquad \forall (r,t) \tag{26}$$

$$n^P_{fgrt}, n^S_{fgrt} \geq 0 \qquad \forall (f,g,r,t) \tag{27}$$

The objective aims to minimize costs summing inventory holding, demand satisfaction costs over products and periods as the first three components. We adopt the same approach for demand satisfaction costs as Kalay and Taşkın (2020). It is favorable to satisfy a demand rather than unsatisfying regardless of the backlog period length. Hence, the cost associated with unsatisfaction is calculated as $b_j\ (|T| - t + 1)$, which reflects the assumption that demand can be satisfied from an infinite capacity after the planning horizon ends with a corresponding backlog cost associated. Moreover, we additionally include production costs in our model, which are excluded from the objective in single machine instance. The last two terms in the objective correspond to setup costs within a given pattern $p$ and between families $f$ and $g$ over period boundaries respectively. Equation (1) ensures the consistency of demand satisfactions. Equation (2) is the inventory balance constraints. Equation (3) couples variables representing number of days of production allocated in an order for a family to production quantity variables. Order translates into the beginning, middle or ending of a period. Inequality (4) ensures that production quantities in a time period consist a feasible composition within a specific family on a production line. Due to the chemical properties of the glass production, random errors are observed over the flat glass. Depending on the cutting decisions, products of different size and quality combinations can be obtained from the same glass sheet. Depending on the production line characteristics, production amount of a specific size $s$ and quality $q$ is known to not exceed a certain ratio of the total production within a given time period. We refer to Taşkın and Ünal (2009) and Kalay and Taşkın (2020) for further information. Equation (5) ensures allocation of patterns to production lines for each period. Inequalities (6)–(8) serve to model a lower bound for production duration of families that are produced in the middle of a pattern, split into two adjacent periods and a proper upper bound, respectively. Equation (9) formulates production line capacity. Equations (10)–(11) determine starting and ending family within a period. $\theta$ variables indicate whether a changeover is performed from family $f$ to family $g$ at the beginning of period $t$ on each production line, and they are related to $\gamma$ variables with Inequalities (12)–(14). Equation (15) ensures that each production line allocates necessary setup time for color transition. Equations (16) and (17) relate setup time variables for families $(n^S, n^E)$ to period based variables $(F, B)$. Model avoids infeasible family transitions with Equation (18). Equations and Inequalities (19)–(27) define variable domains.

### *3.2. Extended Pattern*

Formulations in Kalay and Taşkın (2020) allocate a pattern from a pre-defined set of patterns feasible in terms of minimum production duration of families involved, to each period of the production line. Moreover, a campaign plan is the sequence of families to be produced on a specific production line with start and end times of setups and production runs of families. A campaign plan is itself, from another point of view, another pattern covering the entire planning horizon as that is also a sequence of families to be executed on the corresponding line. Hence, we define each campaign plan as an *extended* pattern.

We define micro period as a unit amount of time. The idea is to be able to represent the continuous data we need to incorporate into our models in multiples of micro periods. Recall from Section 1 that data with continuous time resolution includes setup times and minimum production duration of families. Similar to the definition of a pattern, *extended* patterns also represent the sequence of families and their respective durations, with the exception that durations are expressed as "number of micro periods". From another perspective, we divide the planning horizon into micro periods, and an *extended* pattern is an ordered representation of family allocations to each one of these micro periods. Figure 5 illustrates four different extended pattern examples for a set of two families $F_1$ and $F_2$ each having 2 micro periods of minimum production duration, and 1 and 3 micro periods of sequence-dependent setup times from $F_1$ to $F_2$ and from $F_2$ to $F_1$ respectively. In our formulation, we define months as macro periods and days as micro periods. Macro periods map to period definition of FTBMV-PM. Hence, FTBMV-PM assigns a pattern to a macro period. Note that the durations of macro and micro periods can be defined based on practical needs. As an example, macro periods can correspond to months and micro periods can correspond to days. Moreover, following the continuous nature of the production line we assume that the first family of the pattern does not require a setup.

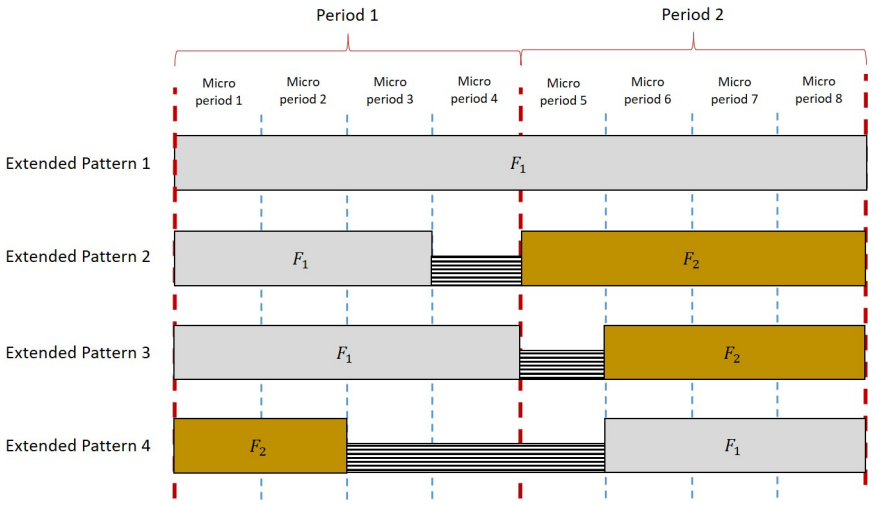


**Figure 5.** Valid extended patterns

### *3.3. Extended pattern based reformulated mathematical model*

An extended pattern covers the entire planning horizon by definition, and assuming such patterns will be constructed ensuring the minimum production duration of families and setup transition feasibility, it enables us to simplify FTBMV-PM. We can associate each production line with an *extended* pattern for the entire planning horizon, hence reformulate the campaign planning problem. We name the reformulation as Extended Pattern Based Campaign Planning Model (CPM-EP). Table 7 lists the symbols used in CPM-EP along with their brief descriptions.

| **Set** | **Description** |
|---|---|
| $J$ | Set of products |
| $R$ | Set of production lines |
| $Q$ | Set of quality groups |
| $S$ | Set of size groups |
| $T$ | Set of time periods |
| $P$ | Set of extended campaign patterns |
| $F$ | Set of product families |
| $P(f)$ | Set of patterns containing family $f$ at least once |
| $F(p)$ | Set of families belonging to pattern $p$ |
| $J(f)$ | Set of products belonging to family $f$ |
| **Parameter** | **Description** |
| $D_{jt}$ | Demand of product $j$ in period $t$ |
| $I_{j(-1)}$ | Beginning inventory of product $j$ |
| $v_{jr}$ | Production speed of product $j$ on production line $r$ |
| $S(j)$ | Index of the size group of product $j$ |
| $Q(j)$ | Index of the quality group of product $j$ |
| $R_{fqsr}$ | Maximum production ratio for quality group $q$ and size group $s$ for family $f$ on production line $r$ |
| $d_{fpt}$ | Number of days family $f$ appears in extended pattern $p$ in period $t$ |
| $h_j$ | Inventory holding cost for product $j$ |
| $b_j$ | Cost of backlogging a demand of product $j$ for a single period |
| $u_{jr}$ | Unit production cost for producing product $j$ on line $r$ |
| $c_{pr}$ | Total setup cost for family order within extended pattern $p$ on production line $r$ |
| **Variable** | **Description** |
| $I_{jt}$ | Inventory of product $j$ at the end of period $t$ |
| $S_{jtk}$ | Satisfied quantity of demand from period $t$ of product $j$ in period $k$ |
| $U_{jt}$ | Unsatisfied quantity of demand from period $t$ of product $j$ |
| $X_{jrt}$ | Production quantity of product $j$ on production line $r$ in period $t$ |
| $\delta_{pr}$ | Binary indicator variable for selection of extended pattern $p$ on production line $r$ |

**Table 7.** Sets and parameters used in CPM-EP

**Model 2.** *Extended Pattern Based Campaign Planning Model (CPM-EP)*

$$min \quad \sum_{j\in J}\sum_{t\in T}\left[h_j\ I_{jt} + b_j\ (|T|-t+1)\ U_{jt} + \sum_{\substack{k\in T\\ k\le t}}(b_j\ (t-k)\ S_{jkt})\right] + \sum_{r\in R}\sum_{j\in J}\sum_{t\in T} u_{jr}X_{jrt} + \sum_{r\in R}\sum_{p\in P} c_{pr}\delta_{pr}$$

*subject to*

$$\sum_{\substack{k\in T\\ k\ge t}} S_{jtk} + U_{jt} = D_{jt} \qquad \forall\ j\in J,\ t\in T \tag{28}$$

$$I_{j(t-1)} + \sum_{r\in R} X_{jrt} - \sum_{\substack{k\in T\\ k\le t}} S_{jkt} = I_{jt} \qquad \forall\ j\in J,\ t\in T \tag{29}$$

$$\sum_{\substack{j\in J(f)\\ Q(j)\le q\\ S(j)\le s}} X_{jrt} \le \sum_{j\in J(f)} X_{jrt}\ R_{fqrs} \qquad \forall\ r\in R,\ f\in F,\ q\in Q,\ s\in S,\ t\in T \tag{30}$$

$$\sum_{p\in P}\delta_{pr} = 1 \qquad \forall\ r\in R \tag{31}$$

$$\sum_{j\in J(f)} v_{jr}X_{jrt} - \sum_{p\in P(f)} d_{fpt}\delta_{pr} = 0 \qquad \forall\ f\in F,\ r\in R,\ t\in T \tag{32}$$

$$I_{jt}, X_{jt}, U_{jt} \ge 0 \quad \forall(j,t) \tag{33}$$

$$S_{jtk} \ge 0 \quad \forall(j,t,k\ge t) \tag{34}$$

$$X_{jrt} \ge 0 \quad \forall(j,r,t) \tag{35}$$

$$\delta_{pr} \in \{0,1\} \quad \forall(p,r) \tag{36}$$

The objective is the same cost minimization only represented with fewer terms since there is no need to represent the incurred setup costs separately for setups within a period and over period boundaries as in Model 1. Equations (28)–(30) are requirement balance, inventory balance and size group quality constraints exactly the same as in Model 1. Equation (31) ensures only a single *extended* pattern is assigned to a production line. Also note that constraints do not include index $t$ since extended patterns cover the entire planning horizon. Equation (32) couples production quantity variables ($X$) with designated duration of corresponding families in the selected extended pattern, ensuring the plan respects the capacity of each production line. Finally, Equations and Inequalities (33)–(36) define variable domains.

Note that Model 2 is a reformulation of Model 1 with only 5 sets of constraints and variable bounds. Moreover, Model 2 has an exponential number of variables due to combination of micro periods and families forming extended patterns. This makes it a candidate for column generation (CG) approach.

## 4. Branch-and-price algorithm

In this section, we explain the algorithm that we propose to solve the campaign planning problem. We first describe the CG specifics in Section 4.1, followed by the modeling of pricing problem as both a binary integer programming formulation and a shortest path problem in Section 4.2. Finally, we explain our initial column set generation, branching and node selection strategies, root node processing approach in addition to upper bound generation in Section 4.3.

### *4.1. Column generation*

The main decision in CPM-EP is the assignment of an extended pattern to machines. Let us refer to the optimal campaign plan in the illustrative numerical example presented in Section 1. Figure 3 shows two extended patterns that are assigned to $R1$ and $R2$, having family order as *YS-RZ* and *RZ* respectively. Let us denote these extended patterns by $p_1$ and $p_2$. Both $p_1$ and $p_2$ qualify as columns of CPM-EP. Corresponding setup costs for $p_2$, namely $c_{p_1(R1)}$ is 0 since it doesn't include any family setup, whereas $p_1$ will incur the setup cost from family YS to RZ. Moreover, we can also deduce the values of $d_{fpt}$ for $p_1$ and $p_2$. In period 1, $p_1$ has 17 days of *YS* production and 6 days of *RZ* production in addition to 5 days of setup. (Note that period duration for period 1 is 28 days). Hence, we have $d_{(YS)p_11} = 17$ and $d_{(RZ)p_11} = 6$. We can use $P' = \{p_1, p_2\}$ with corresponding $c_pr$ and $d_{fpt}$ values.

The number of extended patterns for each machine depends on the number of families that can be produced on the machine, their respective minimum production durations and feasibility of setups between families. Hence, there can be an exponential number of extended patterns, which means in an optimal solution to CPM-EP, most of the corresponding $\delta$ variables will be equal to zero. As a first step of our column generation strategy, we relax the binary variables, $\delta$, in CPM-EP and obtain the linear programming relaxation of restricted master problem as follows:

**Model 3.** *Restricted Extended Pattern Based Campaign Planning Master Model (RCPM-EP)*

$$min \quad \sum_{j\in J}\sum_{t\in T}\left[h_j\ I_{jt} + b_j\ (|T| - t + 1)\ U_{jt} + \sum_{\substack{k\in T\\ k\le t}}(b_j\ (t-k)\ S_{jkt})\right] + \sum_{r\in R}\sum_{j\in J}\sum_{t\in T} u_{jr}X_{jrt} + \sum_{r\in R}\sum_{p\in P'} c_{pr}\delta_{pr}$$

*subject to*

$$(28)\text{–}(32) \tag{37}$$
$$I_{jt}, X_{jt}, U_{jt} \ge 0 \quad \forall(j,t) \tag{38}$$
$$S_{jtk} \ge 0 \quad \forall(j,t,k \ge t) \tag{39}$$
$$X_{jrt} \ge 0 \quad \forall(j,r,t) \tag{40}$$
$$0 \le \delta_{pr} \le 1 \quad \forall(p,r), \quad p \in P' \tag{41}$$

RCPM-EP considers a subset of extended patterns noted $P'$. To generate columns which are not already in $P'$, we are interested in the reduced cost value associated with each such extended pattern. By definition, reduced cost is the amount of necessary improvement in the objective coefficient of the corresponding variable so that the variable becomes a basic variable. Moreover, reduced cost can be calculated by using optimal dual multipliers of the master problem.

We denote the dual variables associated with Equations (31) and (32) by $\pi_r$ and $\mu_{frt}$ respectively. We can find a new column, namely an extended pattern, by checking its reduced cost such that reduced cost with respect to Inequality

(42) is negative,

$$\bar{\pi}_r - \sum_{f \in P(f)} \sum_{t \in T} d_{fpt} \bar{\mu}_{frt} \leq c_{pr} \quad \forall\ r \in R \tag{42}$$

where $\bar{\pi}_r$ and $\bar{\mu}_{frt}$ are optimal dual multipliers from RCPM-EP.

### *4.2. Pricing subproblem*

#### *4.2.1. Pricing subproblem as binary integer programming formulation*

Given an optimal solution of RCPM-EP, we define the pricing problem denoted as (SP($\bar{\pi}_r$,$\bar{\mu}_{frt}$)) for each production line $r$ such that $c_{pr} - \bar{\pi}_r + \sum_{f \in P(f)} \sum_{t \in T} d_{fpt} \bar{\mu}_{frt}$ is minimized and $p$ corresponds to a feasible extended pattern. It needs to respect the minimum production duration of each family as well as sequence dependent setups between families if there are multiple families included. Model 4 shows the binary integer program for pricing subproblem for a given production line $r$.

**Model 4.** *Pricing Subproblem Binary Model (PBM)*

$$min \quad \sum_{m \in M} \sum_{f \in F} \sum_{g \in F} c_{fgr} \gamma_{rfgm} + \sum_{f \in F} \sum_{m \in M} \bar{\mu}_{frt(m)} \theta_{rfm} - \bar{\pi}_r$$

*subject to*

$$\sum_{f \in F} \theta_{rfm} + \alpha_{rm} = 1 \qquad \forall\ m \in M \tag{43}$$

$$\gamma_{rfgm} \leq \theta_{rfm} \qquad \forall\ f \in F, g \in F, g \neq f, m \in M \tag{44}$$

$$\gamma_{rfgm} = \alpha'_m \qquad \forall\ f \in F, g \in F, g \neq f, m, m' \in M, m \leq m' \leq m + ST_{fgr} \tag{45}$$

$$\gamma_{rfgm} = \theta_{rgm'} \qquad \forall\ f \in F, g \in F, g \neq f, m, m' \in M, m + ST_{fgr} \leq m' \leq m + ST_{fgr} + MD_{gr} \tag{46}$$

$$\gamma_{rfgm} \in \{0,1\} \quad \forall (f,g,m) \tag{47}$$

$$\theta_{rfm} \in \{0,1\} \quad \forall (f,m) \tag{48}$$

$$\alpha_{rm} \in \{0,1\} \quad \forall (m) \tag{49}$$

Let us first note that index $m$ correspond to micro periods, and $M$ is the set of all micro periods. Moreover, $t(m)$ maps to the macro period that a given $m$ is in. The binary variables $\theta_{rfm}$ indicate whether machine $r$ is producing family $f$ in $m$, while $\alpha_{rm}$ indicates whether $r$ is in setup state in $m$. Finally, $\gamma_{rfgm}$ variables indicate whether machine $r$ will start setup from family $f$ to family $g$ at the end of $m$. The first term in the objective sums over all setup costs, which is equivalent to $c_{pr}$. The summation of $\theta$ variables in the second term is equivalent to $d_{fpt}$ for a given family $f$. Since each term is multiplied with corresponding optimal dual multipliers $\bar{\mu}$, the objective is equivalent to the objective of (SP($\bar{\pi}_r$,$\bar{\mu}_{frt}$)). Equation (43) ensures that $r$ is either producing a family or in setup state. Inequality (44) ensures that $r$ is indeed producing family $f$ in $m$ so that it can start setup from $f$ to $g$. Equation (45) makes sure that a proper setup time is spent while Equation (46) ensures minimum production duration for each family after setup. Note that to ensure minimum production duration feasibility for the beginning and ending of the planning horizon the model can be adjusted by restricting proper variables to be either 0 or 1. We

note that the number of variables and constraints of PBM can become large quickly with respect to number of families and micro periods.

### *4.2.2. Pricing subproblem as shortest path problem*

We can represent an extended pattern as a path on a special network of the corresponding machine. For each micro period in the planning horizon we create a node for each family $f$. Hence, when a node is in a path it means that the production line $r$ is dedicated to producing products from family $f$ in that micro period. In addition, we create a source and a sink node so that a path maps to a directed flow between them. Arcs of this network is constructed in a way that:

- the minimum production duration of each family is respected
- there exist arcs between family pairs such that a setup is feasible
- when an arc corresponds to a setup, it respects both the setup duration in between families and the minimum production duration of the successor family.

Note that, for arcs corresponding to a setup and minimum duration of the successor family, nodes of the family in related *micro periods* are not in the path but they are in the production plan.

In order to illustrate the idea, let $F_1$ and $F_2$ be two families to be produced on a machine in a planning horizon of 8 micro periods, with 2 micro periods of minimum production duration each. Sequence-dependent setup times from $F_1$ to $F_2$ and from $F_2$ to $F_1$ are 1 and 3 micro periods respectively. We further assume that, each period consists of four micro periods. Figure 6 shows the corresponding network.

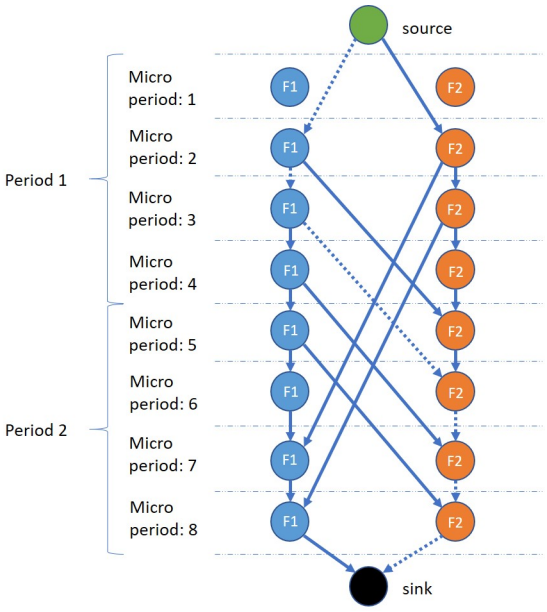


**Figure 6.** An illustrative s-t network

All the paths in this network are valid extended patterns. To further clarify the illustration, we provide in Figure 7 the gantt representation of the path consisting

of arcs in dotted lines. Arc covering micro periods 4 to 6 corresponds to a setup from $F_1$ to $F_2$ on micro period 4, and minimum duration of the successor family $F_2$ on micro periods 6 and 7. Note that minimum production duration of $F_2$ is covered by production in micro periods 5 and 6, however the corresponding arcs, e.g. $F_2$ in micro periods 5 and 6 in Figure 6, are not inbound or outbound to any dashed arc.

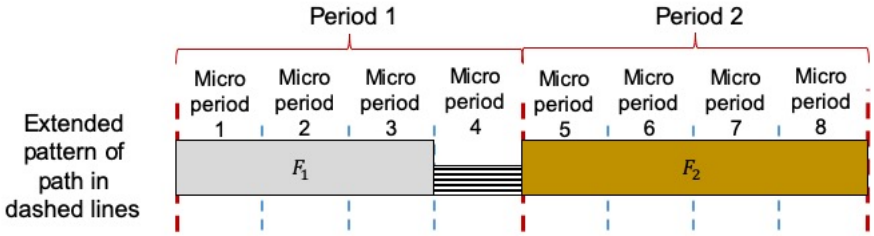


**Figure 7.** Illustration of an s-t network path

Considering our motivation to generate a new column for RCPM-EP with a promising reduced cost defined as the objective function of pricing problem, it is sufficient for us to calculate proper costs on arcs. For outbound arcs of source and inbound arcs of sink node, the associated cost is zero. For all other arcs, the cost is calculated as follows: we determine the duration of production in each period and multiply with the proper dual multiplier of the family associated with *destination node*. Moreover, if family associated with *source node* is different than family of *destination node*, we account for a setup cost and the minimum duration of this *destination* family. Once we calculate the arc costs, the most promising candidate extended pattern is given by the s-t shortest path in this network.

We note that the network is a directed acyclic graph (DAG). Since there is no cycle in a DAG by definition, no negative cost cycle can exist. Hence, shortest paths are well defined, and we can solve it efficiently with topological ordering, in $O(|A|)$ time complexity following from Ahuja, Magnanti, and Orlin (1993), where $|A|$ is the number of arcs in the network. Pricing subproblem (SP($\bar{\pi}_r$,$\bar{\mu}_{frt}$)) can be represented as a DAG and at each iteration we can generate a new column, namely an extended pattern, in polynomial time by calculating proper arc costs. Algorithm 1 illustrates how the algorithm based on topological sort determines the shortest path. Starting from source for each node, the algorithm determines the shortest inbound arc based on label of the origin node of inbound arcs and the corresponding cost on arcs themselves. Once algorithm reaches sink node visiting all nodes exactly once, then it is possible backtrack the determined predecessor and generate the shortest path. We solve the pricing problem with shortest path approach instead of solving BPM.

### *4.3. Algorithm details*

We will apply a B&P algorithm, which focuses on generating columns for tightening an LP relaxation, for the solution of CPM-EP. We defined column generation and pricing problem in Section 4.1 and Section 4.2 respectively. In this section, we will focus on the details of the algorithm.

**Algorithm 1:** Shortest path with topological sort

**CalculateShortestPath** $(G)$
- **inputs :** Network $G$ topologically sorted with proper costs on arcs $(c_a)$
- **output:** Shortest path
- $P, S \leftarrow \emptyset$
- $C \leftarrow 0$
- **foreach** *node* $n \in G$ **do**
  - $L[n] \leftarrow \infty$
- **foreach** *node* $n \in G$ **do**
  - $a \leftarrow \text{argmin}_{a \in I_n} = \{c_a + L[O_a]\}$ where $c_a$ is cost and $O_a$ is origin of arc $a$, $I_n$ is incoming arcs of $n$
  - **if** $a$ is *null* **then**
    - $L[n] = 0$
  - $S[n] \leftarrow$origin node of arc $a$
  - $L[n] \leftarrow$cost of arc $a + L[S[n]]$
- $n \leftarrow$Sink
- **while** $n \neq$ Source **do**
  - $P \leftarrow P \cup n$
  - $n \leftarrow S[n]$
- **return** $P$

*4.3.1. Generating initial set of columns*

Starting the algorithm requires an initial set of columns to be assumed as $P'$ that RCPM-EP will run with. In principal, such a subset can be determined by running constructive heuristics. We generate "unit" extended patterns, which are patterns such that only a single family $f$ is produced on a production line during the entire planning horizon. We generate all possible unit patterns $P_u$ for all resources and start the B&P algorithm using them as initial set of columns. We note that if the number of resources are less than the number of families, then there can be unsatisfied demand for some of the families depending on the capacity. Since we solve the RCPM-EP with relaxed $\delta$ variables, then theoretically it is possible to satisfy all demands with proper $\delta$ values. However, even if there is unsatisfied demand in the solution with initial columns, this is not a problem for the overall algorithm since these initial unit extended patterns are only used to generate initial dual variables to proceed with column generation.

*4.3.2. Branching and node selection strategy*

At each iteration, we solve RCPM-EP with $P'$, followed by solving the pricing subproblem to identify columns to enter the basis for improved objective. Note that we solve the pricing s-t shortest path problem for each production line and add all new columns to $P'$. When the pricing problem is unable to generate a column that will price out, the solution to RCPM-EP is optimal if it is integer feasible. Otherwise, this means some $\delta$ variables are fractional and we need to do branching.

There are fundamental difficulties in applying column generation techniques for linear programming in integer programming solution methods (Lime, Gross-

mann, & Jiao, 2011). It is essential to choose a branching rule which does not increase the complexity of pricing problem solution. Considering conventional branching on variables, it has the potential to destruct the structure of the pricing problem, which is the case for our network representation. Suppose that we obtain a fractional value in the optimal solution to RCPM-EP on the illustrative network in Figure 6 for the extended pattern $p'$, $F_1 - F_1 - F_1 - SETUP - F_2 - F_2 - F_2 - F_2$.

In order to branch on corresponding $\delta$ variable, for the upper branch we can remove all the arcs other than representing $p'$ from the network of the production line. However, for the lower branch, to have $\delta_{p'} = 0$, if we remove the arcs not included in $p'$, then we also cut off some other feasible extended patterns as well, of which $F_1 - F_1 - F_1 - F_1 - F_1 - F_1 - F_1 - F_1$ is an example in the example on Figure 6. Such branching destructs the structure of the pricing problem since we will need a special algorithm, which in the worst case requires enumerating all the paths leading to exponential complexity in $|N|$. Hence, we need to adopt another branching rule. An intuitive approach is to define a branching strategy that will correspond to removal of some arcs and/or nodes from the network.

We define a branching strategy based on family nodes in the network of each production line $r$. When the CG is unable to generate new columns, we calculate, for each arc on the network, a weight index such that for each arc it is the sum of all the extended pattern variables with positive value. This provides us with a "superposition" of these patterns. Then starting from the source node, we start tracking paths with positive values. At some node, the paths will need to be separated which provides reasonable branching point on the network.

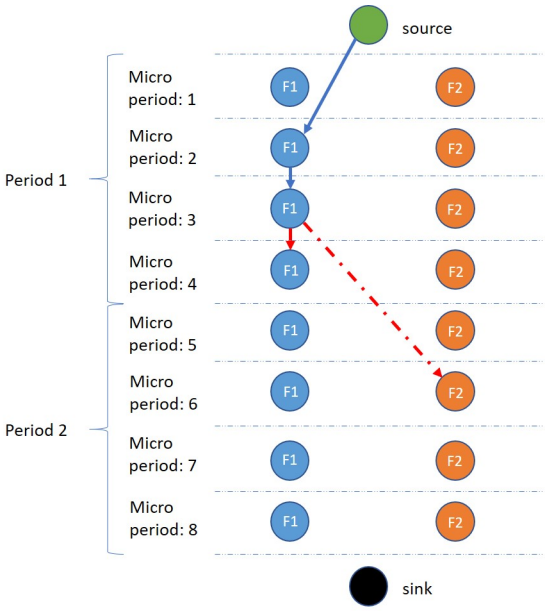


**Figure 8.** An illustrative branching instance

Figure 8 illustrates such an instance. There are two different paths and each one reaches $F_1$ in micro period 2 from source, continues on $F_1$ onto micro period 3. At the end of micro period 3, there are two possible next nodes to reach: $F_1$ in micro period 4 and $F_2$ in micro period. Suppose that weights on corresponding

arcs are 0.2 and 0.8. We select the arc with highest value, which in this case is the arc from $F_1$ in micro period 3 to $F_2$ in micro period 6. Our branching decision is then on upper branch, the network should have to provide a path being in $F_2$ producing state in micro period 4 from now on throughout the algorithm. On the lower branch, the network should not provide any path being in $F_2$ producing state in micro period 4.

Note that we do the branching in the adjacent micro period of the last common node and that being in the setup for destination family is admitted as producing that family, in this case $F_2$. Another important note is that, in order to avoid redundant search in the tree, we add B&P node of one single production line at each iteration. We select the resource to create the branches such that the resource has its separation at the earliest micro period. If there are multiple resource separation at same micro period, then we select the resource with highest number of families that can be produced on.

At each node, we simply activate or deactivate a node from the network corresponding to a family in a micro period, which we call the branching rule. In practical terms, this corresponds to deciding whether to produce a family in a specific micro period on a production line or not. Considering the branching instance described above, we can intuitively remove the node $F_2$ in micro period 4 for the lower branch so that the machine produces $F_1$. However, this approach is incomplete since there exist arcs which implicitly adds up to the undesired state. In Figure 6 which is the complete network, arc from $F_1$ in micro period 2 to node $F_2$ in micro period 5 means that the machine is producing $F_2$ in micro period 4. Hence, it requires to carefully account for all nodes and arcs and remove as necessary to ensure the desired state of the machine on the lower and upper branches.

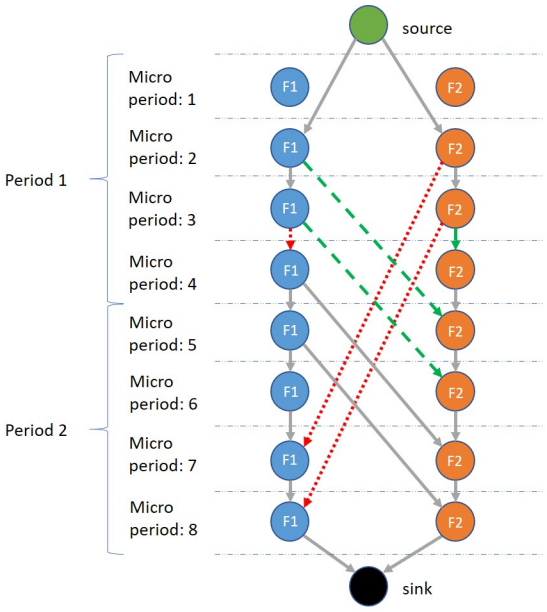


**Figure 9.** Color coded arcs related to producing $F_2$ in micro period 4

Dashed arcs (in green) in Figure 9 means the machine will produce $F_2$ in micro period 4 and dotted arcs (in red) impose the opposite. Solid arcs are unrelated

to the machine state in micro period 4. Figure 10 shows the residual network for upper branch in (a) and for lower branch in (b). Note that, we also removed $F_1$ node in the upper and $F_2$ node in the lower branch along with their inbound and outbound arcs.

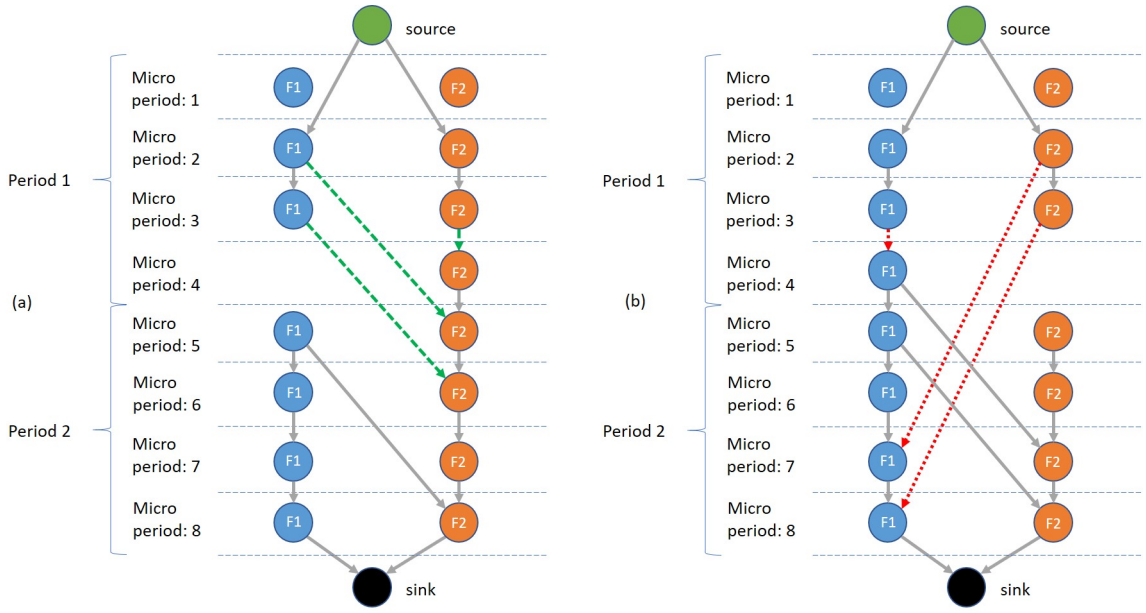


**Figure 10.** Upper (a) and Lower (b) branches related to producing $F_2$ in micro period 4

Since we employ all the related branching rules to the network at each B&P node to be processed , any generated column throughout the algorithm respects all of these rules. Hence, going down the B&P search tree, we will not get any infeasibility and will work on more and more sparse s-t networks. We apply three most common strategies for the node selection from the search tree: depth-first-search (DFS), breadth-first-search (BFS) and best-bound-selection (BBS).

### *4.3.3. Upper bound generation*

Upper bound generation is an essential part of B&P algorithms in the sense that it potentially provides an incumbent solution and enables pruning of nodes with lower bounds greater than the upper bound. Hence, it is necessary to come up with an improved approach such that it is capable of balancing running time efficiency with quality of the solutions found.

As a heuristic approach we propose to run the CPM-EP with $P'$ such that $P'$ contains all generated columns if we are processing the root node. Otherwise, $P'$ will consist of extended patterns with positive $\delta$ variable value in the latest solution of RCPM-EP. In this way, the heuristic can search within a larger set of columns while processing the root node and a limited subset while processing any other node.

We limit the execution time of this upper bound generation heuristic by 300 seconds. Since the model in the root node contains, in general, a much larger set of columns compared to other nodes, it might terminate without proven

optimality. The CPM-EP on other nodes in the B&P tree reaches to an optimal solution within 30 seconds according our preliminary observations.

#### *4.3.4. Adaptive Root node processing*

Root node is by definition the first node in the search tree. Intuitively, we start with $P'$ generated as described in Section 4.3.1. However, using *unit* extended patterns as $P'$ is questionable in the sense that the CG algorithm might converge slowly to the state where it cannot generate any column, namely finishes processing the root node. It can be possible for CG to converge faster with a different initial set of $P'$. Making use of micro-period concept can provide another means of generating initial set of patterns such that the root node processing performance is better with respect to processing time.

We defined micro-periods as a unit amount of time multiplies of which can represent problem data, especially the input data in continuous resolution. The longer the micro-period length is, the more lightweight will be the shortest path network structure. Consequently, the CG algorithm may converge faster. On the other hand, having longer micro-periods will result in discrepancy in the sense that the input data is not incorporated into the model fully. We suggest an approach to dynamically change the micro-period length throughout the processing of root node, which we call *adaptive micro-period length*.

We set an initial positive integer $k$ value, and for each $k$, we process the root node setting the micro-period length to $2^k$ unit amount of time. Note that, we use unit extended patterns as $P'$ for the initial $k$ value. Once CG converges, we keep the incumbent solution in $P'$, remove all other patterns, reduce $k$ by 1 and iterate on until $k = 0$. For each value of $k$, the network is a *'shrunk'* version of the one with $k+1$. Hence, an extended pattern, a path in the network context, can be represented in either of the networks. Note also that, for $k = 0$, we have the original problem but this time with a different initial column set than unit extended patterns. This structure allows us to adaptively change the sensitivity of the model to input data and obtain initial solutions that will help CG convergence faster. We name the B&P algorithm employing adaptive root node processing as Adaptive B&P.

## 5. Computational experiments

In this section, we give details about numerical results from running the proposed formulations. We implemented formulations with C# language of the .NET Framework and used commercial solver CPLEX (12.8) for experiments. We executed all experiments on a PC with Intel Core i7-8750H CPU 2.20 GHz and 32 GB RAM.

### *5.1. Data set and problem instances*

The data used in the numerical experiments, similar to Kalay and Taşkın (2020), is based on real life data provided by a major float glass manufacturer in Turkey. Hence, the data is realistic in terms of production, setup and cost perspective. Kalay and Taşkın (2020) define three different family structures, which have effect on the complexity of the problem. Having color and coating in the structure

is described as the most complex case, and we will use this structure in our experiments.

The difficulty in problem stems from the decision on pattern allocation to resources. In FTBMV-PM, we need to make this decision for each resource and macro period. In B&P for CPM-EP, we progressively generate the extended patterns. The number resources and the number of macro periods effect the number of all possible extended patterns. Hence, it is important to test the performance of B&P for CPM-EP and FTBMV-PM with datasets ranging in terms of number of resources and periods.

We have two base datasets each containing 1482 unique products of different color, size, quality, coating, thickness and packaging type attributes. Two datasets differ from each other with respect to the number of machines; one has 3 production lines whereas the other has 5 lines. In each case, all the machines are unrelated in terms of production speed for products. For each of these base datasets, we have 5 different demand scenarios, and for each one of the demand scenarios, we solve the problem for 5 different planning horizon length. In detail, we solve the instances for 4, 6, 8, 10 and 12 periods.

We assume the micro period length as days and note that it can be modified according to data or practical needs. Moreover, we adjusted the minimum production durations and setup times to be allocated over period boundaries between adjacent patterns in FTBMV-PM.

### *5.2. Preliminary tests on node selection strategy*

We define different node selection strategies, namely DFS, BFS and BBS, in Section 4.3.2 to be employed throughout the solution. We first run some preliminary tests to determine whether a strategy outperforms the others. In order to capture performance against different complexities, we randomly selected 3 instances for combinations of 3 and 5 machines with 4, 8 and 12 periods. We used CPLEX as linear programming (LP) solver. We limit the overall solution time to 3600 seconds for all instances. Note that we do not employ the adaptive root node processing explained in Section 4.3.4 in our preliminary tests.

Tables 8 and 9 show the outputs of preliminary test results with 3 and 5 parallel machines respectively. Note that *Run* column is a representation for the run instance with the example that "m:3-p:4-d:3" stand for instances with 3 parallel machines (m), 4 macro periods (p) and demand scenario (d) 3. For each test instance and node selection strategy, we report the optimality gap ("Gap" column) and the number of nodes explored in the search tree ("Nodes" column). We calculate Gap as $|ObjBound - ObjVal|/|ObjVal|$ where $ObjBound$ is lower bound and $ObjVal$ is incumbent solution objective value.

Results show that on 3 machine tests, BFS outperforms the other strategies in 7 out of 9 runs. DFS and BBS strategies each are able to provide the best outputs in 1 instance. On 5 machine tests on the other hand, the outlook is exactly on the contrary. DFS outperforms other methods in 7 out of 9 runs with BBS and BFS having 1 best performance each. First conclusion is that regarding the node selection strategy in the solution approach we propose, BBS is outperformed by the other methods. Moreover, for different machine configurations we observe significantly better performing strategies. Finally, in all test instances BBS has the least number of nodes process from the search tree. Hence, for the remainder

| | BBS | | DFS | | BFS | |
|---|---|---|---|---|---|---|
| **Run** | **Gap** | **Nodes** | **Gap** | **Nodes** | **Gap** | **Nodes** |
| m:3-p:4-d:1 | 11.26% | **9052** | 11.26% | 8932 | **9.29%** | 5216 |
| m:3-p:4-d:3 | 10.58% | **8840** | 10.58% | 8438 | **9.31%** | 3733 |
| m:3-p:4-d:5 | **10.52%** | **9900** | 10.86% | 9454 | 11.56% | 5341 |
| m:3-p:4-avg | 10.79% | **9264** | 10.90% | 8941 | **10.05%** | 4763 |
| m:3-p:8-d:2 | 14.69% | **4485** | 14.69% | 4247 | **13.21%** | 137 |
| m:3-p:8-d:3 | 13.32% | 1139 | 13.32% | **1184** | **13.31%** | 151 |
| m:3-p:8-d:4 | 15.43% | **1163** | 15.43% | 1106 | **11.66%** | 132 |
| m:3-p:8-avg | 14.48% | **2262** | 14.48% | 2179 | **12.73%** | 140 |
| m:3-p:12-d:2 | 14.15% | 11 | 14.15% | **21** | **14.07%** | 11 |
| m:3-p:12-d:3 | 13.90% | 17 | 13.90% | **22** | **13.20%** | 6 |
| m:3-p:12-d:5 | 14.36% | 6 | **13.45%** | **8** | 14.36% | 1 |
| m:3-p:12-avg | 14.14% | 11 | **13.83%** | **17** | 13.88% | 6 |

**Table 8.** Test results on node selection strategy on 3 machines

| | BBS | | DFS | | BFS | |
|---|---|---|---|---|---|---|
| **Run** | **Gap** | **Nodes** | **Gap** | **Nodes** | **Gap** | **Nodes** |
| m:5-p:4-d:1 | 5.16% | 6766 | 4.31% | **9331** | **3.67%** | 1404 |
| m:5-p:4-d:4 | 10.16% | 505 | **2.89%** | **8639** | 5.84% | 1545 |
| m:5-p:4-d:5 | 5.63% | 410 | **4.05%** | **8408** | 6.38% | 2148 |
| m:5-p:4-avg | 6.98% | 2560 | 3.75% | **8793** | **5.30%** | 1699 |
| m:5-p:8-d:1 | 6.40% | 286 | **6.32%** | **1000** | 6.57% | 157 |
| m:5-p:8-d:2 | 7.41% | 10 | **5.39%** | **252** | 7.09% | 134 |
| m:5-p:8-d:4 | 7.57% | 14 | **3.87%** | **364** | 6.58% | 147 |
| m:5-p:8-avg | 7.13% | 103 | 5.19% | **539** | **6.75%** | 146 |
| m:5-p:12-d:1 | 10.04% | 1 | **10.04%** | 1 | 10.04% | 1 |
| m:5-p:12-d:4 | 13.61% | 1 | **13.61%** | 1 | 13.61% | 1 |
| m:5-p:12-d:5 | **11.86%** | 1 | 12.00% | 1 | 12.00% | 1 |
| m:5-p:12-avg | 11.84% | 1 | 11.88% | 1 | **11.88%** | 1 |

**Table 9.** Test results on node selection strategy on 5 machines

of our numerical experiments, we will employ BFS in 3 machine instances and DFS in 5 machine instances.

### *5.3. Numerical results*

Considering dataset and algorithm settings described in Section 5.1, our numerical experiments consist of 200 instances for the B&P algorithm. For the comparison, we take FTBMV-PM runs for all datasets, which count up to additional 100 instances. Similar to our approach in preliminary tests explained in Section 5.2, we limit the overall solution time to 1-hour for all instances. For each unique dataset instance, we also solve the model with FTBMV-PM with a 1-hour time limit. Moreover, since our B&P algorithm is implemented to work on single thread, to have a fair comparison we limit the number of threads to be used by CPLEX as one.

Tables 10 and 11 show the outputs of B&P for CPM-EP and FTMBV-PM for 3 and 5 parallel machines respectively. We report the average optimality gap calculated with the same formula defined in Section 5.2 from all run instances for FTBMV-PM, B&P and Adaptive B&P. Note that *Run* column is a smart representation indicating number of machines with $m$ and number of periods with $p$ as explained in previous Section. With the increase in number of periods, the algorithms obtain solution with higher optimality gaps. Overall, B&P algorithms outperform FTBMV-PM for both 3 and 5 machine instances. Amongst B&P algorithms proposed, adaptive approach consistently performs better than classical approach in 3 machine instances. On the other hand, in 5 machine instances, adaptive approach outperforms classical approach in 3 out of 5.

| Run | FTBMV | B&P | B&P Adaptive |
|---|---|---|---|
| m:3-p:4 | **4.62%** | 10.24% | 10.19% |
| m:3-p:6 | 26.39% | 11.51% | **10.00%** |
| m:3-p:8 | 38.57% | 13.63% | **11.57%** |
| m:3-p:10 | 48.18% | 13.32% | **12.44%** |
| m:3-p:12 | 49.25% | 13.97% | **13.05%** |

**Table 10.** Average gap FTBMV-PM compared to B&P algorithms on 3 machines

| Run | FTBMV | B&P | B&P Adaptive |
|---|---|---|---|
| m:5-p:4 | 6.26% | 4.77% | **4.76%** |
| m:5-p:6 | 72.86% | 5.82% | **5.64%** |
| m:5-p:8 | 82.96% | **5.79%** | 6.50% |
| m:5-p:10 | 88.63% | **7.01%** | 7.44% |
| m:5-p:12 | 88.69% | 12.03% | **8.54%** |

**Table 11.** Average gap FTBMV-PM compared to B&P algorithms on 5 machines

Tables 12 and 13 present explicit results for all run instances comparing FTBMV-PM with B&P with classical root node processing B&P with adaptive root node processing for 3 parallel machines and for 5 parallel machines respectively. Note that, with our approach, we need to have the root node processing finished in order to provide an optimality gap. In cases where the B&P algorithms require more than the given time limit to provide an optimality gap, we note the duration as *'time'*.

We compare the results similar to the approach described in Kalay and Taşkın (2020). An algorithm outperforms the other if it obtains a solution with lower

| | FTBMV-PM | | B&P | | B&P Adaptive | |
|---|---|---|---|---|---|---|
| **Run** | **Gap** | **Time** | **Gap** | **Time** | **Gap** | **Time** |
| m:3-p:4-d:1 | **0.00%** | 469 | 9.29% | 3600 | 12.25% | 3600 |
| m:3-p:4-d:2 | **0.00%** | 340 | 10.02% | 3600 | 10.02% | 3600 |
| m:3-p:4-d:3 | **0.00%** | 604 | 9.31% | 3600 | 9.31% | 3600 |
| m:3-p:4-d:4 | **0.00%** | 370 | 11.49% | 3600 | 11.44% | 3600 |
| m:3-p:4-d:5 | **0.00%** | 287 | 11.56% | 3600 | 10.96% | 3600 |
| m:3-p:4-d:6 | **8.42%** | 3600 | 11.43% | 3600 | 9.56% | 3600 |
| m:3-p:4-d:7 | 7.61% | 3600 | **7.05%** | 3600 | 7.15% | 3600 |
| m:3-p:4-d:8 | 17.67% | 3600 | 9.85% | 3600 | **9.33%** | 3600 |
| m:3-p:4-d:9 | **8.06%** | 3600 | 11.38% | 3600 | 10.72% | 3600 |
| m:3-p:4-d:10 | **4.40%** | 3600 | 11.07% | 3600 | 11.14% | 3600 |
| m:3-p:6-d:1 | **2.19%** | 3600 | 10.04% | 3600 | 10.04% | 3600 |
| m:3-p:6-d:2 | **1.78%** | 3600 | 11.74% | 3600 | 11.74% | 3600 |
| m:3-p:6-d:3 | **2.54%** | 3600 | 10.08% | 3600 | 10.08% | 3600 |
| m:3-p:6-d:4 | **3.18%** | 3600 | 8.62% | 3600 | 8.62% | 3600 |
| m:3-p:6-d:5 | **1.80%** | 3600 | 10.84% | 3600 | 10.84% | 3600 |
| m:3-p:6-d:6 | 74.27% | 3600 | 12.63% | 3600 | **10.55%** | 3600 |
| m:3-p:6-d:7 | 52.35% | 3600 | 13.55% | 3600 | **12.37%** | 3600 |
| m:3-p:6-d:8 | 18.15% | 3600 | 12.09% | 3600 | **10.13%** | 3600 |
| m:3-p:6-d:9 | 53.49% | 3600 | 12.03% | 3600 | **8.03%** | 3600 |
| m:3-p:6-d:10 | 54.15% | 3600 | 13.50% | 3600 | **11.17%** | 3600 |
| m:3-p:8-d:1 | **4.12%** | 3600 | 14.32% | 3600 | 11.21% | 3600 |
| m:3-p:8-d:2 | **3.29%** | 3600 | 13.21% | 3600 | 11.96% | 3600 |
| m:3-p:8-d:3 | **4.52%** | 3600 | 13.31% | 3600 | 12.14% | 3600 |
| m:3-p:8-d:4 | **4.76%** | 3600 | 11.66% | 3600 | 11.41% | 3600 |
| m:3-p:8-d:5 | **5.93%** | 3600 | 12.07% | 3600 | 11.83% | 3600 |
| m:3-p:8-d:6 | 84.09% | 3600 | 12.73% | 3600 | **10.98%** | 3600 |
| m:3-p:8-d:7 | 84.51% | 3600 | 14.93% | 3600 | **10.71%** | time |
| m:3-p:8-d:8 | 58.50% | 3600 | 16.24% | 3600 | **12.05%** | 3600 |
| m:3-p:8-d:9 | 58.42% | 3600 | 15.49% | 3600 | **10.48%** | 3600 |
| m:3-p:8-d:10 | 77.59% | 3600 | **12.40%** | 3600 | 12.95% | 3600 |
| m:3-p:10-d:1 | **7.17%** | 3600 | 12.58% | 3600 | 13.78% | time |
| m:3-p:10-d:2 | **11.43%** | 3600 | 13.64% | 3600 | 12.20% | time |
| m:3-p:10-d:3 | **10.80%** | 3600 | 15.39% | 3600 | 14.29% | time |
| m:3-p:10-d:4 | **8.48%** | 3600 | 12.61% | 3600 | 11.46% | time |
| m:3-p:10-d:5 | **10.71%** | 3600 | 12.16% | 3600 | 12.60% | time |
| m:3-p:10-d:6 | 86.72% | 3600 | 13.17% | 3600 | **11.62%** | time |
| m:3-p:10-d:7 | 86.59% | 3600 | 13.68% | 3600 | **12.14%** | time |
| m:3-p:10-d:8 | 86.63% | 3600 | 14.51% | 3600 | **12.56%** | time |
| m:3-p:10-d:9 | 86.67% | 3600 | 12.64% | 3600 | **11.54%** | time |
| m:3-p:10-d:10 | 86.58% | 3600 | 12.76% | 3600 | **12.24%** | time |
| m:3-p:12-d:1 | 13.62% | 3600 | 13.10% | 3600 | **12.54%** | time |
| m:3-p:12-d:2 | **10.06%** | 3600 | 14.07% | time | 12.69% | time |
| m:3-p:12-d:3 | **9.81%** | 3600 | 13.20% | 3600 | 12.94% | time |
| m:3-p:12-d:4 | **10.82%** | 3600 | 15.84% | time | 12.89% | time |
| m:3-p:12-d:5 | **8.21%** | 3600 | 14.36% | time | 15.03% | time |
| m:3-p:12-d:6 | 87.98% | 3600 | **13.68%** | 3600 | 13.77% | time |
| m:3-p:12-d:7 | 88.00% | 3600 | 12.46% | 3600 | **11.56%** | time |
| m:3-p:12-d:8 | 87.97% | 3601 | 14.43% | 3600 | **11.56%** | time |
| m:3-p:12-d:9 | 88.04% | 3600 | **13.60%** | 3600 | 13.61% | time |
| m:3-p:12-d:10 | 88.02% | 3600 | 15.01% | 3600 | **13.92%** | time |

**Table 12.** FTBMV-PM compared to B&P and B&P Adaptive on 3 machines

| | FTBMV-PM | | B&P | | B&P Adaptive | |
|---|---|---|---|---|---|---|
| **Run** | **Gap** | **Time** | **Gap** | **Time** | **Gap** | **Time** |
| m:5-p:4-d:1 | 5.21% | 3600 | 4.31% | 3600 | **4.31%** | 3600 |
| m:5-p:4-d:2 | **2.87%** | 3600 | 5.43% | 3600 | 5.35% | 3600 |
| m:5-p:4-d:3 | 7.94% | 3600 | 3.75% | 3600 | **3.75%** | 3600 |
| m:5-p:4-d:4 | 10.86% | 3600 | **2.89%** | 3600 | 2.95% | 3600 |
| m:5-p:4-d:5 | 5.29% | 3600 | 4.05% | 3600 | **4.05%** | 3600 |
| m:5-p:4-d:6 | 7.76% | 3600 | **4.46%** | 3600 | 4.84% | 3600 |
| m:5-p:4-d:7 | **2.29%** | 3600 | 3.55% | 3600 | 3.37% | 3600 |
| m:5-p:4-d:8 | **5.56%** | 3600 | 7.00% | 3600 | 7.67% | 3600 |
| m:5-p:4-d:9 | 7.48% | 3600 | **5.45%** | 3600 | 5.63% | 3600 |
| m:5-p:4-d:10 | 7.35% | 3600 | 6.82% | 3600 | **5.66%** | 3600 |
| m:5-p:6-d:1 | 75.09% | 3600 | 6.17% | 3600 | **6.17%** | 3600 |
| m:5-p:6-d:2 | 83.09% | 3600 | 5.07% | 3600 | **5.07%** | 3600 |
| m:5-p:6-d:3 | 77.27% | 3600 | 6.66% | 3600 | **5.58%** | 3600 |
| m:5-p:6-d:4 | 77.29% | 3600 | 5.90% | 3600 | **5.89%** | 3600 |
| m:5-p:6-d:5 | 28.56% | 3600 | 5.44% | 3600 | **5.44%** | 3600 |
| m:5-p:6-d:6 | 73.46% | 3600 | 6.77% | 3600 | **6.75%** | 3600 |
| m:5-p:6-d:7 | 79.03% | 3600 | 5.09% | 3600 | **4.96%** | 3600 |
| m:5-p:6-d:8 | 78.77% | 3600 | **4.61%** | 3600 | 4.64% | 3600 |
| m:5-p:6-d:9 | 78.36% | 3600 | 6.09% | 3600 | **5.86%** | 3600 |
| m:5-p:6-d:10 | 77.72% | 3600 | 6.40% | 3600 | **4.82%** | 3600 |
| m:5-p:8-d:1 | 82.05% | 3600 | 6.32% | 3600 | **6.29%** | 3600 |
| m:5-p:8-d:2 | 83.02% | 3600 | **5.39%** | 3600 | 6.32% | 3600 |
| m:5-p:8-d:3 | 82.02% | 3600 | **7.00%** | 3600 | 8.17% | 3600 |
| m:5-p:8-d:4 | 79.06% | 3600 | **3.87%** | 3600 | 5.62% | 3600 |
| m:5-p:8-d:5 | 81.80% | 3600 | **6.48%** | 3600 | 7.39% | 3600 |
| m:5-p:8-d:6 | 85.65% | 3600 | **6.22%** | 3600 | 6.39% | 3600 |
| m:5-p:8-d:7 | 85.61% | 3600 | **5.34%** | 3600 | 5.69% | 3600 |
| m:5-p:8-d:8 | 82.82% | 3600 | **5.85%** | 3600 | 6.94% | 3600 |
| m:5-p:8-d:9 | 81.96% | 3600 | **5.80%** | 3600 | 5.98% | 3600 |
| m:5-p:8-d:10 | 85.64% | 3600 | **5.68%** | 3600 | 6.21% | 3600 |
| m:5-p:10-d:1 | 87.36% | 3600 | **7.31%** | 3600 | 7.40% | time |
| m:5-p:10-d:2 | 87.38% | 3600 | 7.01% | 3600 | **5.95%** | time |
| m:5-p:10-d:3 | 87.34% | 3600 | 7.42% | 3600 | **7.00%** | time |
| m:5-p:10-d:4 | 87.31% | 3600 | **6.72%** | 3600 | 7.06% | time |
| m:5-p:10-d:5 | 87.37% | 3600 | **7.54%** | 3600 | 8.08% | time |
| m:5-p:10-d:6 | 87.38% | 3600 | **6.21%** | 3600 | 8.20% | time |
| m:5-p:10-d:7 | 87.37% | 3600 | **6.12%** | 3600 | 6.68% | time |
| m:5-p:10-d:8 | 87.40% | 3600 | 9.30% | 3600 | **7.63%** | time |
| m:5-p:10-d:9 | 87.39% | 3600 | **6.61%** | 3600 | 7.76% | time |
| m:5-p:10-d:10 | 100.00% | 3600 | **5.82%** | 3600 | 8.63% | time |
| m:5-p:12-d:1 | 88.66% | 3600 | 10.04% | time | **8.04%** | time |
| m:5-p:12-d:2 | 88.69% | 3600 | 11.66% | time | **9.29%** | time |
| m:5-p:12-d:3 | 88.65% | 3600 | 10.80% | time | **8.43%** | time |
| m:5-p:12-d:4 | 88.63% | 3600 | 13.61% | time | **8.45%** | time |
| m:5-p:12-d:5 | 88.77% | 3600 | 12.00% | time | **7.87%** | time |
| m:5-p:12-d:6 | 88.71% | 3600 | 13.46% | time | **7.19%** | time |
| m:5-p:12-d:7 | 88.78% | 3600 | 13.65% | time | **11.08%** | time |
| m:5-p:12-d:8 | 88.65% | 3600 | 12.71% | time | **7.16%** | time |
| m:5-p:12-d:9 | 88.60% | 3600 | **9.17%** | time | 9.28% | time |
| m:5-p:12-d:10 | 88.72% | 3600 | 13.20% | time | **8.64%** | time |

**Table 13.** FTBMV-PM compared to B&P and B&P Adaptive on 5 machines

optimality gap. In 3 machine instances we note FTBMV-PM outperforms B&P results in 26 out of 50 instances. We observe that FTBMV-PM's performance is rather unstable. In 3 machine and 8 periods instances for instance although the average optimality gap is approximately 38% whereas the explicit instances vary from 3.29% to 84.51%. In 5 machines instances, B&P algorithms perform better than FTBMV-PM, in 47 out of 50 instances. Moreover, the adaptive root node processing is able to provide better optimality gaps in 26 instances whereas the classical approach provides the best gap in 21 instances. We note that, with the increase in complexity of the run instance through increase in number of resources and periods, the B&P algorithms tend to take more time to process, however generating better solutions. The algorithm we propose, according to the numerical experiments, is capable of providing good quality solutions in more complex instances when MIP formulation FTBMV-PM fails to do so. When there is a smaller number of parallel machines on the other hand, FTBMV-PM and B&P approaches seem to be both applicable. However, FTBMV-PM's performance appears to be unstable with high volatility in optimality gap.

Tables 14 and 15 present the root node processing times and optimality gap before branching for B&P and B&P Adaptive methods. B&P Adaptive consistently takes longer to process the root node compared to B&P. Each micro period length processed in B&P Adaptive contributes to increased root node processing time. On the other hand, we observe that the optimality gap obtained by B&P Adaptive consistently outperforms B&P method.

| | **B&P** | | **B&P Adaptive** | |
|---|---|---|---|---|
| **Run** | **Gap** | **Time** | **Gap** | **Time** |
| m:5-p:4 | 19.53% | 9.59 | 16.30% | 87.98 |
| m:5-p:6 | 15.19% | 50.85 | 13.66% | 358.10 |
| m:5-p:8 | 16.69% | 322.23 | 12.32% | 1074.63 |
| m:5-p:10 | 13.64% | 1421.45 | 12.44% | time |
| m:5-p:12 | 15.11% | 2308.95 | 13.21% | time |

**Table 14.** Average gap and root node processing times of B&P algorithms on 3 machines

| | **B&P** | | **B&P Adaptive** | |
|---|---|---|---|---|
| **Run** | **Gap** | **Time** | **Gap** | **Time** |
| m:5-p:4 | 9.12% | 136.03 | 7.59% | 422.86 |
| m:5-p:6 | 9.17% | 159.05 | 6.63% | 556.47 |
| m:5-p:8 | 7.97% | 944.59 | 6.60% | 2209.75 |
| m:5-p:10 | 7.88% | 3238.11 | 7.31% | time |
| m:5-p:12 | 12.22% | 3600.00 | 8.54% | time |

**Table 15.** Average gap and root node processing times of B&P algorithms on 5 machines

## 6. Conclusion

In this paper we studied the parallel machine campaign planning problem under sequence dependent family setups and co-production in the process industry. We proposed a B&P algorithm to solve the problem and designed some variations differing in terms of root node processing and node selection strategy. We compared our algorithms with an extension to a previous study, namely FTBMV-PM. With the runs using a realistic dataset, FTBMV-PM however outperforms proposed algorithms in instances smaller in number of machines. Our algorithms,

on the other hand, are able to provide better results when the instances get more complex with the increase in number of parallel machines and periods.

As a future research direction, we can extend the research by including multiple facilities and multiple bill-of-material (BOM) levels. Moreover, as stated in Section 5.3, B&P currently has a limitation to be running on a single thread, which exposes extensibility to parallelism. Also, improving a more sophisticated initial column set can also help in accelerating the root node processing time, which does not improve significantly with adaptive root node processing.

## Acknowledgement(s)

Z. Caner Taşkın's research was partially supported by Turkish Science Academy BAGEP award.

## References

Ahuja, R. K., Magnanti, T. L., & Orlin, J. B. (1993). *Network flows: Theory, algorithms, and applications*. Prentice hall.

Akker, J., Hoogeveen, J., & Kempen, J. (2012). Using column generation to solve parallel machine scheduling problems with minmax objective functions. *Journal of Scheduling*, *15*, 1-10.

Almada-Lobo, B., Klabjan, D., Maria, Carravilla, A., & Oliveira, J. F. (2010). Multiple machine continuous setup lotsizing with sequence-dependent setups. *Computational Optimization and Applications*, *47*, 529–552.

Ağralı, S. (2012). A dynamic uncapacitated lot-sizing problem with co-production. *Optimization Letters*, *6*, 1051–1061.

Bektur, G., & Saraç, T. (2019). A mathematical model and heuristic algorithms for an unrelated parallel machine scheduling problem with sequence-dependent setup times, machine eligibility restrictions and a common server. *Computers and Operations Research*, *103*, 46–63.

Camargo, V., Toledo, F., & Almada-Lobo, B. (2012). Three time-based scale formulations for the two-stage lot sizing and scheduling in process industries. *Journal of the Operational Research Society*, *63*, 1613–1630.

Camargo, V., Toledo, F., & Almada-Lobo, B. (2014). Hops – hamming-oriented partition search for production planning in the spinning industry. *European Journal of Operational Research*, *234*, 266–277.

Chen, Z.-L., & Powell, W. B. (1999). Solving parallel machine scheduling problems by column generation. *INFORMS Journal on Computing*, *11*, 78-94.

Farahani, P., Grunow, M., & Günther, H. O. (2012). Integrated production and distribution planning for perishable food products. *Flexible Services and Manufacturing*, *24*, 28–51.

Fleischmann, B., & Meyr, H. (1997). The general lotsizing and scheduling problem. *OR Spectrum*, *19*, 11–21.

Florian, M., K. Lenstra, J., & H. G. Rinnooy Kan, A. (1980). Deterministic production planning: Algorithms and complexity. In *Management science* (p. 669-679).

Gaudreault, J., Frayret, J.-M., Rousseau, A., & D'Amours, S. (2011). Combined planning and scheduling in a divergent production system with co-production: A case study in the lumber industry. *Computers and Operations Research*, *38*, 1238–1250.

Ghirardi, M., & Ameiro, A. (2019). Matheuristics for the lot sizing problem with back-ordering, setup carryovers, and non-identical machines. *Computers and Industrial Engineering*, *127*, 822–831.

Glass, G. P. F. (2019). Float glass - gold plus glass [Computer software manual]. Retrieved from https://goldplusgroup.com/float-glass/ (accessed in September 2020)

Guimarães, L., Klabjan, D., & Almada-Lobo, B. (2014). Modeling lot sizing and scheduling problems with sequence dependent setups. *European Journal of Opreations Research*, *239*, 644–662.

Hinder, O., & Mason, A. J. (2017). A novel integer programing formulation for scheduling with family setup times on a single machine to minimize maximum lateness. *European Journal of Operational Research*, *262*, 411–423.

James, R. J., & Almada-Lobo, B. (2011). Single and parallel machine capacitated lotsizing and scheduling: New iterative MIP-based neighborhood search heuristics. *Computers and Operations Research*, *38*, 1816–1825.

Joly, M., Moro, L. F. L., & Pinto, J. M. (2002). Planning and scheduling for petroleum refineries using mathematical programming. *Brazilian Journal of Chemical Engineering*, *19*, 207–228.

Kalay, S., & Taşkın, Z. C. (2020). Single machine campaign planning under sequence dependent family setups and co-production. *Journal of the Operational Research Society*. Retrieved from https://doi.org/10.1080/01605682.2020.1772016

Kopanos, G., Puigjaner, L., & Maravelias, C. (2011). Production planning and scheduling of parallel continuous processes with product families. *Industrial Engineering and Chemistry Research*, *50*, 1369–1378.

Kramer, A., Iori, M., & Lacomme, P. (2019). Mathematical formulations for scheduling jobs on identical parallel machines with family setup times and total weighted completion time minimization. *European Journal of Operational Research*. Retrieved from https://doi.org/10.1016/j.ejor.2019.07.006

Lime, R., Grossmann, I., & Jiao. (2011). Long-term scheduling of a single-unit multi-product continuous process to manufacture high performance glass. *Computers and Checmial Engineering*, *35*, 554–574.

Miegeville, N. (2005). *Supply chain optimization in the process industry. methods and case-study of the glass industry.* (Unpublished doctoral dissertation). Ecole Centrale, Paris, Paris.

Pessoa, A., Uchoa, E., Poggi, M., & De Freitas Rodrigues, R. (2010). Exact algorithm over an arc-time-indexed formulation for parallel machine scheduling problems. *Mathematical Programming Computation*, *2*, 259-290.

Rafiei, R., Nourelfath, M., Gaudreault, J., Santa-Eulalia, L. A. D., & Bouchard, M. (2015). Dynamic safety stock in co-production demand-driven wood remanufacturing mills: A case study. *International Journal of Production Economics*, *165*, 90–99.

Schaller, J. (2014). Minimizing total tardiness for scheduling identical parallel machines with family setups. *Computers and Industrial Engineering*, *72*, 274–281.

Seeanner, F., Almada-Lobo, B., & Meyr, H. (2013). Combining the principles of variable neighborhood decomposition search and the fix&optimize heuristic to solve multi-level lot-sizing and scheduling problems. *Computers and Operations Research*, *40*, 303–317.

Susarla, N., & Karimi, I. (2011). Integrated campaign planning and resource allocation in batch plants. *Computers and Chemical Engineering*, *35*, 2990–3001.

Taşkın, Z. C., & Ünal, A. T. (2009). Tactical level planning in float glass manufacturing with co-production, random yields and substitutable products. *European Journal of Operational Research*, *199*(1), 252–261.

Vieira, M., Pinto-Varela, T., Moniz, S., Barbosa-Póvoa, A. P., & Papageorgiou, L. G. (2016). Optimal planning and campaign scheduling of biopharmaceutical processes using a continuous-time formulation. *Computers and Chemical Engineering*, *91*, 422–444.

Xiong, X., Zhou, P., Yin, Y., Cheng, T. C. E., & Li, D. (2019). An exact branch-and-price algorithm for multitasking scheduling on unrelated parallel machines. *Naval Research Logistics*, *66*, 502-516.

Yin, Y., Chen, Y., Qin, K., & Wang, D. (2018). Two-agent scheduling on unrelated parallel machines with total completion time and weighted number of tardy jobs criteria. *Journal*

*of Scheduling*, *22*, 315-333.