Heuristic algorithm for a WIP projection problem at finite capacity in semiconductor manufacturing
Emna Mhiri, Fabien Mangione, Mireille Jacomino, Philippe Vialletelle, Guillaume Lepelletier

To cite this version:
Emna Mhiri, Fabien Mangione, Mireille Jacomino, Philippe Vialletelle, Guillaume Lepelletier. Heuristic algorithm for a WIP projection problem at finite capacity in semiconductor manufacturing. IEEE Transactions on Semiconductor Manufacturing, Institute of Electrical and Electronics Engineers, 2018, 31, pp.62-75. hal-01682760

HAL Id: hal-01682760
https://hal.archives-ouvertes.fr/hal-01682760
Submitted on 12 Jan 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Heuristic algorithm for a WIP projection problem at finite capacity in semiconductor manufacturing

Emna Mhiri, Fabien Mangione, Mireille Jacomino, Philippe Vialletelle, and Guillaume Lepelletier

Abstract—In this paper, we propose a heuristic approach for fixing work-in-progress (WIP) projection issues in the semiconductor industry especially for High Mix Low Volume (HMLV) facilities. The considered problem consists of estimating the start and end dates for each remaining process step of the production lots in the WIP and anticipating the fab loading taking into account the constraints of the maximum throughput of machines considered as capacity constraints and customer delivery commitments. The objective being to guarantee on-time delivery, we focus on minimizing the total weighted tardiness (TWT). We have formulated the problem into a mixed-integer programming (MIP) and we have empirically shown its computational intractability. Due to the computational intractability using actual production data, a heuristic algorithm is proposed. It is an iterative finite capacity planning system that considers as inputs lots due dates and equipment capabilities and capacities. The performance of the heuristic is assessed using industrial instances. It turns out that it achieves targeted objectives with satisfactory results in terms of quality of the solution and computation time.

Index Terms—WIP projection; finite capacity planning; semiconductor industry; mixed integer programming; iterative algorithm.

I. INTRODUCTION

Work-in-progress (WIP) projection is a mid-term capacity planning activity. The objective is to compute a mid-term target schedule in order to drive factory execution, to anticipate production issues and to calculate net demand and net resource capacities. In our study, the outcome is a weekly-released schedule that depicts the start and end dates of each remaining processing step as well as the expected workload accumulated on each equipment per time bucket over the planning horizon.

In this study, the WIP projection problem is considered in one of the most dynamic industries in the world, the semiconductor industry. The semiconductor manufacturing process is extremely complex and constantly innovating. The considered wafer production plant is a High Mix Low Volume (HMLV) production line: there are several hundreds of products, different technologies and heterogeneous toolsets i.e. collections of nonidentical multi-purpose parallel machines (or tools). Moreover, typical semiconductor fabrication processes require several hundreds of different steps. As, for obvious reasons, HMLV fabs cannot multiply machines, their production flows are re-entrant: the same machine can process products at different stages of their fabrication. This also means that according to the decisions taken on the production line, products may experience various cycle times depending on the priority given either to a given product, to satisfy customer demand, or to a certain technological level for the purpose of line balancing. Hence, the capacity planning issue is difficult to solve and it is particularly more complex than in other industries [1].

Semiconductor manufacturing is composed of four major phases: wafer fabrication (fab), wafer probe, assembly, and final test. Wafer fabrication, often referred to as "front end", represents the most complicated, expensive and time-consuming phase of all four stages [2]. In this phase, hundreds of circuits are layered through successive operations on a silicon wafer. The manufacturing process in wafer fabs involves a highly complex sequence of processing operations which can be classified into various types, as for example: oxidation and thermal treatment, film deposition, planarization, photolithography, etching and ion implantation. These operations are repeated for each layer of circuitry on the wafer. Figure 1 presents a simplified view of the wafer fabrication process.

Fig. 1: Wafer fabrication Process. [1]

Each operation shown in Figure 1 can include multiple elementary steps (cleaning, process, measurement). The total number of steps per flow typically ranges between 400 and 800 for current production technologies and up to 1200 steps for latest generations. Some of the processing steps in a flow are performed on individual wafers, others on groups of wafers (lots), and still others on groups of lots (batches).

Steps performed on individual wafers or lots of wafers are referred to as serial steps, while those performed on groups of lots are called batch steps. A lot is generally composed of
25 wafers, while a typical batch contains up to six lots. In the considered case study, the lot requires 8 to 10 weeks to be processed. Steps are executed on more than one hundred workstations called “toolset” [1]. Due to flow re-entrance, lots visit the same toolset more than once during the manufacturing process.

For each step, the wafer has to be processed on various types of tools using a well-defined recipe. The recipe contains the detailed instructions to be used at the machine level in order to proceed the intended physical transformations or measurements. The identification of the candidate tools to be used is made through qualification of recipes on the tools. However, in HMLV fabs, because of multiple differences in hardware and software configurations, hence variety of recipes to be used, it is not possible to qualify all recipes on every machine. Qualification is one of the characteristics of the HMLV semiconductor manufacturing. It determines the processing authorization of a product on a machine. It acts like an eligibility constraint that allows production volume allocation of a product to a machine. It is known also as the process capability constraint [3].

Besides, each toolset has an identified throughput considered as its capacity which refers to its upper loading threshold under a given product mix condition. To establish a feasible production schedule over a planning horizon of several weeks, thus requires to consider capability and capacity constraints. Moreover, as for other industries, semiconductor manufacturing facilities must respect customers delivery commitments to survive in competitive business environments. For HMLV fabs, actual cycle time is widely spread and skewed due to large variability of numerous sources: equipment heterogeneity, product priorities, low redundancy, steps qualifications, etc. It is then crucial to consider also variable cycle times while defining a production plan. In practice, fab’s historical data and various applications of the queuing theory are often used.

In this paper, a mixed integer program (MIP) and a heuristic algorithm are proposed to project current inventory and new wafer starts throughout the remaining processing sequence, taking into account all the cited constraints. The objective is to establish a feasible midterm schedule, in a fast execution time (less than 5 minutes, the required computation time of capacity planners of the industrial partner), while minimizing lots delivery delays and optimizing workload balance among all toolsets. This study is applied to the Crolles 300 mm wafer fab of STMicroelectronics. Thus, data from actual production process are collected and used to evaluate the performance of the developed approaches.

This paper is organized as follows. This section introduces the main characteristics of the considered industrial environment. In section 2, some background on existing related capacity planning problems is provided. In section 3, the problem is stated and the MIP formulation is presented. The proposed iterative heuristic algorithm is explained in section 4. Next, in section 5, experiments conducted and analyses carried out are discussed. Finally, section 6 draws conclusions and provides suggestions for future work.

II. PREVIOUS RELATED WORK

As the semiconductor industry is considered as one of the most complex manufacturing processes, many researchers have paid attention to the capacity planning problems encountered in this environment.

The various problems investigated have considered different phases of the manufacturing process of integrated circuits, different constraints, different methods and techniques used for capacity planning and different performance measures. Mönch et al. [1], Uzsoy et al. [2], [4] and Gupta et al. [5] have mentioned in their reviews different capacity planning techniques used in the semiconductor environment which can be classified in infinite and finite capacity planning techniques. They can also be divided, according to the length of the planning horizon, into long-term (strategic), mid-term (tactical) and short-term (operational) planning tools.

Among the methods used for capacity planning, classical techniques were successfully used in many industries especially for tactical and operational production planning, such as Material Requirement Planning (MRP) developed by Orlicky [6], Manufacturing Resource Planning (MRPII) [7], Just In Time (JIT) [8] and Theory Of Constraints (TOC) [9]. The application of these traditional techniques for capacity planning in semiconductor industry presents some shortcomings. Indeed, it is proven that MRP method can be inefficient and may produce unrealistic production schedules when used in field applications. It ignores capacity constraints and assumes fixed cycle times ([10], [11], [12], [13]). However, in semiconductor facilities, cycle times depend on many factors, such as machine utilization rate, lot size, inventory and dispatching rules, and are thus variable. Either shortcoming above leads to infeasible production schedules, fluctuating workloads over time and significant users effort to adjust the plans. The JIT technique proves its strengths [14]; however, it presents some limitations in the high-mix low-volume production systems. It seems to be more suitable for a repetitive production environment with stable demand and low product mix [15]. The TOC seems an efficient capacity planning technique in semiconductor industry [16] but it considers only bottleneck resources and it can not deal with changes in the bottlenecks.

In addition to these classical industrial methods, authors use discrete event simulation models, queuing theory, linear programming and heuristics for capacity planning applied to semiconductor industry. Discrete event simulation is often used for capacity planning decisions in wafer fabs [17] in order to evaluate the performance of production planning strategies ([18], [19], [20], [21], [22]). Indeed, discrete-event simulation is considered as the only practical method that explicitly calculates the cycle time as a function of resource availability and production rate. The simulation model can be used also to determine bottlenecks under a given product mix and to make strategic decisions concerning equipment purchase [23]. However, simulation models used for capacity planning in the semiconductor industry present some severe limitations. Their set-up is very time-consuming due to the volume and often complexity of the data required for the models involved. Moreover, these models do not provide a
Concerning queuing network models, Shanthikumar et al. [26] presented a survey of the different applications of queuing theory for semiconductor manufacturing systems. They recognized that in spite of fast computing time compared with simulation models, the accuracy of classical queuing models is not satisfactory due to the complexity of the semiconductor manufacturing process.

The linear programming (LP) approach is widely applied to specific issues encountered in capacity planning for the semiconductor industry. Mixed-integer programming (MIP) models are developed for strategic planning in order to maximize the profit ([27], [28], [29]) or to minimize the machine tool operating costs, new tool acquisition costs, and inventory holding costs taking into account capacity constraints [30]. A good source of previous work related to more strategic capacity planning is provided by Geng and Jiang [31].

LP (sometimes in combination with discrete-event simulation) is also used to solve medium-term finite capacity planning problems. The work of Hung and Leachman [32] is an example of such an approach. Leachman [33] used LP for production planning and presents a corporate capacity planning model, which includes multiple facilities integrated with the production process. Habla et al. [34] suggested a MIP formulation to determine completion time targets for the lots on bottleneck steps. Bermon et al. [35] introduced a linear programming model to analyze the capacity of large and complex manufacturing production lines.

Due to the intractability of LP models, they are generally combined with heuristics such as genetic algorithms [36] or decomposition techniques as Benders [37] or Lagrangian relaxation ([30], [34]) to reduce execution time. Besides, approximate methods have also been widely used to develop either infinite or finite capacity planning systems for the semiconductor industry.

Infinite capacity planning systems are developed to estimate the future loading of equipment in order to identify bottleneck resources and to balance the loading of each production resource over the planning horizon ([38], [39], [40]).

Bearing in mind the importance of capacity constraints, many authors developed finite capacity planning systems using algorithmic approaches. Fargher et al. [41] used a beam-search algorithm in combination with backtracking steps for lot release and for the determination of schedules in an aggregated sense.

Horiguchi et al. [42] proposed an algorithm that estimates the start and finish date of each job scheduled on each critical resource: their algorithm considers the available time for all the feasible combinations of time bucket and critical resource, and it reduces the available time whenever a new production order is added to the schedule. This approach, due to the high aggregation level in modeling resources and relationships, might lead to orders overlapping on the same resource in the same time bucket (i.e., infeasible plans).

Habenicht and Mönch [43] used also a beam-search algorithm to determine planned start and completion dates for the macro operations (sets of consecutive process steps) of a lot.

Chua et al. [44] developed an intelligent multi-constraint finite capacity-based lot release system. This system has been designed, developed and implemented to solve the lot release problems in a discrete manufacturing environment with a huge product mix and multiple capacity constraints.

In this study, we are interested in operations research related (or mathematical) optimization approaches and we consider a medium-term finite capacity planning problem, applied to a semiconductor production line. So far, the literature review has pointed out that the debate about this problem is still open, and the proposed approaches by several authors still have some limits. Table II presents a taxonomy of studies considering the same problem and using operations research solving tools. Steps cycle time can be either defined as a fixed input by the proposed approach or a variable output of the procedure. Capability constraints are relevant, since they can be embedded or not in the proposed procedure. Lots due dates, relevant as well, can be considered as input parameters or not. Finally, the model can be tested via data generated by authors (random instances) or through data from real-life production systems (real case).

Whilst capacity and cycle time are tightly linked one another through the Little’s law [45], cycle time is considered by most approaches as a fixed input parameter. Moreover, some methods ignore capability constraints thus leading to infeasible production plans. Finally, the applicability in field to real-life companies has not been reported for all the anterior studies.

Furthermore, Table II presents, for each study, its algorithmic and operational objectives. As one can notice, the main issues treated in the existing studies are generally limited to dispatching rules and release control policies which are outside the scope of this paper.

In the literature, there are few studies considering the WIP projection problem in the semiconductor industry ([46], [47], [48]). In these works, authors consider different objectives and do not take into account all the cited constraints.

Kim and Leachman [46] proposed a LP formulation and a decomposition heuristic method to determine net demand and net resource capacities taking into account capacity constraints. They tested their approaches using random data. Lee et al. [47] employed deterministic linear programming techniques for the WIP projection problem in the wafer fab, that explicitly considers the variable cycle time. Govind and Fronkowski [48] consider WIP projection problem to measure production performance at IBM’s 300 mm wafer fab by computing productivity and WIP targets at infinite capacity. As one can see, even if some papers tackle similar planning problems, none of the already proposed models explicitly address our specific problem.

The research work outlined here tried to overcome some of the limits above: the proposed finite capacity planning algorithm does not consider fixed steps cycle time, it takes into account lots due dates and it has been tested in a real-life industrial context. Furthermore, it meets the key requirement of semiconductor industrials, consisting on fast computing of feasible production plans (in five minutes at most on a personal computer) to facilitate "what-if” analysis.
III. MATHEMATICAL FORMULATION OF THE PROBLEM

A. Problem description

The WIP projection problem may be defined as follows. A set of lots \( l \in \{1, \ldots, L\} \), composed of \( Q_l \) wafers each, is considered. For each lot \( l \), it remains an identified number of steps \( S_l \) to be processed on a time horizon discretized in \( T \) periods \( t \in \{1, \ldots, T\} \) of equal length \( P_t \). Each lot of a weight \( w_l \) indicating its priority, has a release date \( r_l \) and a due date \( d_l \).

The performance measurement to be minimized in this problem is total weighted tardiness \( TWT \). \( TWT \) is a measure that incurs a penalty for each lot that finishes processing after its promised delivery date. This penalty increases with the magnitude of the tardiness, and therefore schedules that minimize the weighted (by lot priority) sum of penalties provide good on-time delivery performance, whereas higher values of total weighted tardiness indicate that many important lots are not being delivered on time. Indeed, a processing schedule will provide a completion time, \( C_l \), for each lot. The tardiness, \( T_l \), of lot \( l \) is then defined as \( T_l = \max(0, C_l - d_l) \). The total weighted tardiness of the lot \( l \) \( (WTT_l) \) is defined as \( WTT_l = (w_l \times T_l) \). Total weighted tardiness computes the weighted sum of tardiness values: \( TWT = \sum_l WTT_l \).

Each remaining step \( s_{l} \in \{1, \ldots, S_l\} \) of the lot \( l \) is processed on one or several qualified toolsets \( i \in \{1, \ldots, I\} \). The quantity of wafers of a lot \( l \) assigned to the toolset \( i \), processing the step \( s_{l} \) during the period \( t \), is denoted \( a_{s_l,i,t} \). It has a waiting time \( w_{s_l,i,t} \) and it consumes a unit processing time \( p_{s_l,i,t} \) on each of its qualified processing toolsets. It also has a start date \( s_{s_l,i,t} \) and an end date \( e_{s_l,i,t} \) (Figure 2). Each toolset \( i \) has a finite capacity \( C_{i,t} \), which gives the maximal loading \( L_{i,t} \) over a period \( t \).

Table I summarizes the notation.

![Figure 2: Problem description.](image)

<table>
<thead>
<tr>
<th>Indices</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( L )</td>
<td>Number of lots</td>
</tr>
<tr>
<td>( l = 1..L )</td>
<td>Lot index</td>
</tr>
<tr>
<td>( S_l )</td>
<td>Number of remaining steps of lot ( l )</td>
</tr>
<tr>
<td>( s_{l} = 1..S_l )</td>
<td>Lot’s step index</td>
</tr>
<tr>
<td>( i = 1..I )</td>
<td>Toolset index</td>
</tr>
<tr>
<td>( T )</td>
<td>Number of time buckets</td>
</tr>
<tr>
<td>( t = 1..T )</td>
<td>Period index</td>
</tr>
</tbody>
</table>

Parameters Description

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( P_t )</td>
<td>Length of period ( t )</td>
</tr>
<tr>
<td>( Q_l )</td>
<td>Quantity of wafers of lot ( l )</td>
</tr>
<tr>
<td>( r_l )</td>
<td>Release date of lot ( l )</td>
</tr>
<tr>
<td>( w_l )</td>
<td>Weight of lot ( l )</td>
</tr>
<tr>
<td>( d_l )</td>
<td>Due date of lot ( l )</td>
</tr>
<tr>
<td>( p_{s_l,i,t} )</td>
<td>Unit processing time of step ( s_{l} ) of lot ( l ) on qualified toolset ( i ), 0 on non-qualified toolset ( i )</td>
</tr>
<tr>
<td>( C_{i,t} )</td>
<td>Capacity of toolset ( i ) in period ( t )</td>
</tr>
<tr>
<td>( a_{s_l,i,t} )</td>
<td>Quantity of wafers of lot ( l ) in step ( s_{l} ) processed by the toolset ( i )</td>
</tr>
</tbody>
</table>

Decision variables Description

<table>
<thead>
<tr>
<th>Decision variables</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s_{s_l,i,t} )</td>
<td>Start date of step ( s_{l} ) of lot ( l )</td>
</tr>
<tr>
<td>( e_{s_l,i,t} )</td>
<td>End date of step ( s_{l} ) of lot ( l )</td>
</tr>
<tr>
<td>( C_l )</td>
<td>Completion date of lot ( l )</td>
</tr>
<tr>
<td>( T_l )</td>
<td>Tardiness of lot ( l )</td>
</tr>
<tr>
<td>( L_{i,t} )</td>
<td>Loading of toolset ( i ) in period ( t )</td>
</tr>
<tr>
<td>( y_{s_l,i,t} )</td>
<td>( = s_{s_l,i,t} ) if the step ( s_{l} ) of lot ( l ) is released in period ( t ), 0 otherwise</td>
</tr>
<tr>
<td>( x_{s_l,i,t} )</td>
<td>( = 1 ) if step ( s_{l} ) of lot ( l ) is processed in period ( t ), 0 otherwise</td>
</tr>
</tbody>
</table>

notation presented above, the MIP is as follows:

\[
\begin{align*}
\min & \quad \sum_l w_l T_l \\
\text{s.c.} & \quad s_{s_l,i,t} \geq r_l & \quad l = 1, \ldots, L \\
& \quad s_{s_l,i,t} + \sum_i p_{s_l,i,t} \times a_{s_l,i,t} \times x_{s_l,i,t} = e_{s_l,i,t} & \quad s_l = 1, \ldots, S_l, l = 1, \ldots, L \\
& \quad s_{s_l,i,t} \geq e_{s_l-1,i,t} & \quad s_l = 2, \ldots, S_l, l = 1, \ldots, L \\
& \quad \sum_t y_{s_l,i,t} = s_{s_l,i,t} & \quad s_l = 1, \ldots, S_l, l = 1, \ldots, L \\
& \quad \sum_t x_{s_l,i,t} = 1 & \quad s_l = 1, \ldots, S_l, l = 1, \ldots, L \\
& \quad C_l = e_{S_l,i,t} & \quad l = 1, \ldots, L \\
& \quad T_l = C_l - d_l & \quad l = 1, \ldots, L \\
& \quad T_l \geq 0 & \quad l = 1, \ldots, L \\
& \quad t \times P_t \times x_{s_l,i,t} \leq y_{s_l,i,t} & \quad s_l = 1, \ldots, S_l, l = 1, \ldots, T \\
& \quad (t + 1) \times P_t \times x_{s_l,i,t} > y_{s_l,i,t} & \quad s_l = 1, \ldots, S_l, l = 1, \ldots, T - 1 \\
& \quad L_{i,t} = \sum_l \sum_{s_l \in S_l} p_{s_l,i,t} \times x_{s_l,i,t} \times a_{s_l,i,t} & \quad i = 1, \ldots, I, t = 1, \ldots, T \\
& \quad L_{i,t} \leq C_{i,t} & \quad i = 1, \ldots, I, t = 1, \ldots, T \\
& \quad x_{s_l,i,t} \in \{0, 1\} & \quad s_l = 1, \ldots, S_l, l = 1, \ldots, L, t = 1, \ldots, T
\end{align*}
\]

B. Mixed-Integer Program

In this subsection, an appropriate MIP formulation is presented for the multi-product, multi-period and multi-resource capacity planning problem. The proposed MIP is similar to LP capacity planning models that can be found in standard textbooks, with some variations and extensions. Using the
<table>
<thead>
<tr>
<th>Approach</th>
<th>Reference</th>
<th>Objectives</th>
<th>Constraints and assumptions</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Algorithmic</td>
<td>Operational</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Linear programming</td>
<td>Hung and Leachman [32], Bermon et al. [35]</td>
<td>Maximize the profit</td>
<td>Determine wafer release quantities</td>
<td>✓</td>
</tr>
<tr>
<td>Algorithms / heuristics</td>
<td>Leachman [33]</td>
<td>Maximize the profit</td>
<td>Generate capacity-feasible start and out schedules</td>
<td>✓ ✓ ✓</td>
</tr>
<tr>
<td></td>
<td>Habla et al. [34]</td>
<td>Minimize total weighted tardiness</td>
<td>Determine completion time targets for the bottleneck steps</td>
<td>✓ ✓ ✓ ✓</td>
</tr>
<tr>
<td></td>
<td>Fargher et al. [41]</td>
<td>Reduce cycle time and the variance of cycle time</td>
<td>Determine the work to release into the factory at any time</td>
<td>✓ ✓ ✓</td>
</tr>
<tr>
<td></td>
<td>Horiguchi et al. [42]</td>
<td>Estimate the start and finish date of each job scheduled on each critical resource</td>
<td>Improve delivery performance and system predictability</td>
<td>✓ ✓</td>
</tr>
<tr>
<td></td>
<td>Habenicht and Mönch [43]</td>
<td>Determine the start date and the end date of each operation of the lot</td>
<td>Establish a feasible production plan</td>
<td>✓ ✓ ✓</td>
</tr>
<tr>
<td></td>
<td>Chua et al. [44]</td>
<td>Compute orders release dates for semiconductor back end assembly</td>
<td>Solve lot release problem</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Our study</td>
<td>Minimize total weighted tardiness</td>
<td>Establish a feasible target production plan</td>
<td>✓ ✓ ✓ ✓</td>
</tr>
</tbody>
</table>

TABLE II: Taxonomy of the literature of finite capacity planning approaches applied to semiconductor industry.
The objective function (1) minimizes the total weighted tardiness (TWT). The MIP constraints can be classified in two kinds: temporal constraints ((2)...(11)) and cumulative constraints (constraints (12)-(13)). Constraints (2) define the start date of the first remaining step for each lot. The end date of each remaining step of each lot is computed using constraints (3). Constraints (4) present precedence constraints of processing steps. Constraints (5) guarantee that each remaining step of each lot is released once. Constraints (6) verify that each remaining step of each lot is processed once during the planning horizon. Constraints (7) define the lots completion date. Constraints (8) and (9) compute the tardiness for each lot. Constraints (10) and (11) indicate that each remaining step of each lot is processed in one period. Constraints (12) calculate the workload accumulated by each toolset over each period taking into account the qualification of the toolset to the processed step and the quantity of wafers assigned to the considered toolset. Constraints (13) are the capacity constraints. Constraints (14) are the binary constraints for the decision variable.

The mathematical model presented above has been solved by ILOG CPLEX solver. Experiments were run on an Intel® Core™ i5 PC running a 2.7 GHz processor and 4 GB of RAM. Tests have been performed on 30 randomly generated instances of the problem in order to highlight the main characteristics of the industrial data and to maintain a certain degree of generality in order to preserve all the difficulty of the problem. Indeed, based on the observation made in the literature, we identified seven important problem parameters which could affect the performance of the proposed approach: number of lots (L), maximum number of remaining steps for each lot (max S_i), number of toolsets (I), length of the planning horizon (T), lots steps unit processing times (p_{s,t,i}), lots due dates (d_l) and machines capacities (C_{t,i}).

We consider the cases of 3, 5, 10, 20, 100 and 300 parallel toolsets with a fixed capacity corresponding to the maximum equipment utilization rate which is equal to 100%. The lots weights w_l are chosen from a uniform distribution over (0,1). The lots release dates r_l and lots quantity of wafers Q_l are supposed equal to 0 and 25 for all lots, respectively. The range of lots due dates d_l and steps unit processing times p_{s,t,i} is extracted from real data. d_l are ranging from 1 to 210 days relative to the release date and p_{s,t,i} range between 0.0005 and 0.5 hours. The planning horizon is set to 24 periods (weeks). Table III presents the different tests parameters generating 30 instances.

Optimal results were obtained in reasonable execution time while testing the MIP on instances of reduced size. Further increasing the size of the tested instances (up to about 4000 steps plan), the resolution of MIP was halted as it required a very large amount of time and computer memory (Figure 3). Indeed, the whole real problem presents 70,742,400 constraints and 69,371,200 variables. It corresponds to a WIP composed of 24 periods (weeks). Thus, the size of real instances is obviously too large to be solved using the proposed MIP (Figure 3).

### TABLE III: Summary of tests parameters

<table>
<thead>
<tr>
<th>Problem parameter</th>
<th>Values used</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of lots (L)</td>
<td>2, 3, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 200, 240, 1000, 1700, 2000</td>
</tr>
<tr>
<td>Maximum number of remaining steps of lot l (max S_l)</td>
<td>1, 2, 5, 6, 8, 10, 20, 30, 40, 50, 60, 100, 150, 200, 250, 680</td>
</tr>
<tr>
<td>Number of toolsets (I)</td>
<td>3, 5, 10, 20, 100, 300</td>
</tr>
<tr>
<td>Number of time buckets (T)</td>
<td>24</td>
</tr>
<tr>
<td>Weight per lot (w_l)</td>
<td>Uniform (0,1)</td>
</tr>
<tr>
<td>Lots release dates (r_l)</td>
<td>0</td>
</tr>
<tr>
<td>Lots due dates (d_l)</td>
<td>r_l+\lfloor1..210\rfloor</td>
</tr>
<tr>
<td>Lots quantity of wafers (Q_l)</td>
<td>25</td>
</tr>
<tr>
<td>Steps unit processing times (p_{s,t,i})</td>
<td>[0.0005..0.5]</td>
</tr>
</tbody>
</table>

Fig. 3: Limits of MIP resolution.

From the empirical evidence on the computational difficulties in getting optimal schedule considering lots due dates and capacity constraints, it is obvious that the problem of WIP projection applied to the real case study will be computationally intractable. Furthermore, Garey and Johnson [49] highlighted in their study that production planning, capacity planning and scheduling problems in complex job shops like semiconductor manufacturing are known as strongly NP-hard problems. This has motivated us to develop a heuristic algorithm for the research problem considered in this study to provide near optimal solutions and/or efficient solution in a reasonable time. The proposed heuristic algorithm is presented in the next section.

### IV. HEURISTIC ALGORITHM

An alternative methodology for the above problem should be accurate and, at the same time, fast and small enough to be stored and implemented in a mainframe or work station computer system. Bearing this in mind, a heuristic approach for WIP projection problem in HMLV semiconductor manufacturing line has been developed. It is an iterative algorithm composed of three main modules: (i) WIP projection at infinite capacity, (ii) workload accumulation and capacity analysis and (iii) workload and capacity balancing. The algorithm is executed by iterations on periods of the planning horizon. The principle of iterative running of the algorithm is inspired from the literature [50] and the detailed scheduling in the commercial ERP/APS. For each defined period, WIP projection module estimates the evolution of the WIP, lot by lot, based on lots due dates. Then, workload accumulation module calculates the expected equipment loading. In case of toolsets over-saturation i.e. the loading of toolsets exceeds their maximal
capacity, workload and capacity balancing module is employed to reduce toolsets loading by shifting their affected steps to subsequent periods. The following sections will detail the three major modules. Figure 4 depicts the flow of the developed system.

![Finite Capacity Planning Algorithm Flow](image)

**Fig. 4: Finite Capacity Planning Algorithm Flow.**

**A. WIP projection module**

The objective of this module is to push lots, one by one, forward along their routes, from their current positions up to their due dates. It also aims to compute over a period the activity required by step to ensure the delivery plan.

For each selected time bucket \( t \) of the planning horizon, this module requires the following data input:

1) WIP status and wafer starts at the beginning of the considered projection period (position \( r_l \), quantity \( Q_l \)),

2) Routing information, including a partition of each route into consecutive steps,

3) Steps unit processing times \( p_{s_l,i} \),

4) Lots due dates \( d_l \) and weights \( w_l \),

5) Flow factor \( X_{factor_{s_l}} \) that reflects possible waiting times between consecutive process steps to achieve the target cycle time, extracted from historical data. It is defined as the step mean cycle time divided by the step raw processing time RPT [51].

WIP projection module includes three steps. Step 1 computes, for each lot, from its position in the route, four parameters which are remaining process time \( RemPT_l \), remaining reference cycle time \( RemRefCT_l \), remaining expected cycle time \( RemExpCT_l \) and cycle time coefficient \( CT_{Coeff_l} \). \( RemPT_l \) is equal to the sum of lot remaining steps unit process time multiplied by lot quantity of wafers \( Q_l \).

\[
RemPT_l = \sum_{s_j=1}^{S_l} \sum_{i=1}^{I} p_{s_l,i} \times Q_l
\]  

(15)

\( RemRefCT_l \) corresponds to the sum of the reference cycle times of lot remaining steps \( RefCT_{s_l} \).

\[
RemRefCT_l = \sum_{s_j=1}^{S_l} RefCT_{s_l}
\]  

(16)

In the industrial context considered, each step has a reference cycle time, extracted from historical data, named \( RefCT_{s_l} \). \( RefCT_{s_l} \) corresponds to the product of the unit step process time with the quantity of lot wafers \( Q_l \) and the flow factor \( X_{factor_{s_l}} \). It is the maximum amount of time that a lot would spend at that step, including waiting and processing times.

\[
RefCT_{s_l} = \sum_{i=1}^{I} p_{s_l,i} \times Q_l \times X_{factor_{s_l}}
\]  

(17)

\( RemExpCT_l \) is equal to the maximum between the difference between the due date and the current time \( t \) and \( RemPT_l \).

\[
RemExpCT_l = \max(d_l - t, RemPT_l)
\]  

(18)

The lot cycle time coefficient \( CT_{Coeff_l} \) identifies the necessary and sufficient speed for lots to achieve their due date according to the reference cycle time. It is equal to the ratio between lot remaining expected cycle time \( RemExpCT_l \) and lot remaining reference cycle time \( RemRefCT_l \).

\[
CT_{Coeff_l} = \frac{RemExpCT_l}{RemRefCT_l}
\]  

(19)

In step 2, the \( RemExpCT_l \) is split on the elementary steps of each lot \( l \) to compute an expected cycle time per step \( ExpCT_{s_l,l} \) which is equal to the product of \( ObjCT_{s_l,l} \) and \( CT_{Coeff_l} \).

\[
ExpCT_{s_l,l} = RefCT_{s_l,l} \times CT_{Coeff_l}
\]  

(20)

Equation (20) gives a rough estimation of queuing time at each step. Hence, waiting time by step \( wt_{s_l,l} \) can be computed:

\[
wt_{s_l,l} = ExpCT_{s_l,l} - I \sum_{i=1}^{I} p_{s_l,i} \times Q_l
\]  

(21)

In step 3, having the waiting time by step, start dates and end dates for all lots remaining steps, decision variables \( x_{l,s_l,t} \), lots completion date and tardiness are computed. Figure 5 illustrates an example of 2 lots with different due dates, having 3 remaining steps each. The first one has an
earlier due date i.e. a higher priority and less $RemExpCT_l$ than the second.

Using the classical projection based on historical data, the two lots have the same distribution of remaining steps waiting times over the planning horizon, independently of their due dates, because they have the same remaining process time $RemPT_l$. However, the proposed projection module allocates steps expected cycle times taking into account lots priorities i.e. due dates. Indeed, there are multiple priority levels of production lots. Production priorities can be divided into two levels according to the urgency of delivery: hot and standard. So, to respect these priorities, the projection module shrinks steps waiting times in order to satisfy the hot lot’s due date. However, for a standard lot, it extends steps waiting times respecting the lot due date.

To further explain the concept of WIP projection, a simple random instance is tested. The considered WIP consists of 10 lots of 25 wafers each, following different routes, and having different due dates. Table IV presents, for each lot, the number of remaining steps, $RemPT_l$, $RemRefCT_l$, $RemExpCT_l$ and $CTCoeff_l$.

![TABLE IV: Data for simple instance](image)

<table>
<thead>
<tr>
<th>Lot</th>
<th>$w_l$</th>
<th>$S_l$ in days</th>
<th>$RemRefCT_l$ in days</th>
<th>$RemExpCT_l$ in days</th>
<th>$CTCoeff_l$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lot 1</td>
<td>0.33</td>
<td>6</td>
<td>1.1</td>
<td>1.6</td>
<td>5</td>
</tr>
<tr>
<td>Lot 2</td>
<td>1</td>
<td>4</td>
<td>0.8</td>
<td>1.1</td>
<td>0.5</td>
</tr>
<tr>
<td>Lot 3</td>
<td>0.5</td>
<td>2</td>
<td>0.25</td>
<td>0.41</td>
<td>1.5</td>
</tr>
<tr>
<td>Lot 4</td>
<td>0.5</td>
<td>8</td>
<td>1.7</td>
<td>2.3</td>
<td>1.5</td>
</tr>
<tr>
<td>Lot 5</td>
<td>0.5</td>
<td>6</td>
<td>1</td>
<td>1.4</td>
<td>1.5</td>
</tr>
<tr>
<td>Lot 6</td>
<td>0.33</td>
<td>4</td>
<td>0.75</td>
<td>1.02</td>
<td>5</td>
</tr>
<tr>
<td>Lot 7</td>
<td>0.5</td>
<td>8</td>
<td>0.86</td>
<td>1.05</td>
<td>1.5</td>
</tr>
<tr>
<td>Lot 8</td>
<td>1</td>
<td>4</td>
<td>0.8</td>
<td>1.05</td>
<td>0.5</td>
</tr>
<tr>
<td>Lot 9</td>
<td>0.5</td>
<td>4</td>
<td>0.8</td>
<td>1.05</td>
<td>1.5</td>
</tr>
<tr>
<td>Lot 10</td>
<td>0.5</td>
<td>6</td>
<td>1.4</td>
<td>1.9</td>
<td>1.5</td>
</tr>
</tbody>
</table>

Fig. 6: Simple instance: Production schedule at infinite capacity.

that have same qualifications and share same recipes. This approach enables to decompose the problem into small sub-problems. It is a linear program used to optimize workload balancing of toolsets, belonging to the same balancing group, over a selected time bucket. The formulation of the linear program, for each balancing group and over each period, is as follows:

**Notations**

Indices:

$B = 1..B$ Number of balancing groups
$R_b$ Number of recipes related to the balancing group $b$
$r = 1..R_b$ Recipe index
$I_b$ Number of toolsets of the balancing group
$I_r$ Number of toolsets qualified for recipe $r$, $I_r \subseteq I_b$
$i = 1..I_b$ Toolset index

B. Workload accumulation and capacity analysis module

After WIP projection, the loading of toolsets, over each considered period $L_{s,t}$, is computed based on the assumption of infinite capacities.

To optimize the computation time, toolsets are distributed in balancing groups. A balancing group is a set of toolsets
Using the above parameters, and decision variables, the linear program seeks to:

- Minimize the total workload of the most loaded toolsets.
- Minimize the total workload of toolsets.
- Maximize the workload of the least loaded toolset among those on which recipe \( r \) is qualified.
- Minimize the workload of the least loaded toolset in the balancing group.

Decision variables:

\[ L_{i,t} \] Loading of toolset \( i \) over period \( t \)

\[ W_{r,i} \] Quantity of wafers produced by toolset \( i \) qualified for recipe \( r \)

\( L_{\text{max}} \) Workload of the most loaded toolset in the balancing group

\( L_{\text{min}} \) Workload of the least loaded toolset in the balancing group

\( L_{r,i} \) Loading, for a given recipe \( r \), of the most loaded toolset among those on which \( r \) is qualified

\( L_{\text{min},r} \) Loading, for a given recipe \( r \), of the least loaded toolset among those on which \( r \) is qualified

Using the above parameters, and decision variables, the linear program formulation can be represented as follows:

\[
\begin{align*}
\text{Minimize} & \quad \alpha \cdot L_{\text{max}} - \beta \cdot L_{\text{min}} + \gamma \cdot \sum_r L_{\text{max},r} - \delta \cdot \sum_i L_{\text{min},i} + \delta \cdot (\sum_b L_{i,t} - L_{\text{min}}) \\
\text{with} & \quad \alpha = I_b^2, \beta = I_b, \gamma = 1, \delta = 1/I_b \\
\text{s.t.} & \quad L_{i,t} = \sum_r p_{r,i} \times W_{r,i} \quad i = 1, \ldots, I_b \\
& \quad \sum_r W_{r,i} = \sum_i \sum_s x_{s,i,t} \times v_{s,i,t,r} \times a_{s,i,t} \\
& \quad r = 1, \ldots, R_b \\
& \quad L_{i,t} \geq L_{\text{min},r} \quad r = 1, \ldots, R_b, i = 1, \ldots, I_r \\
& \quad L_{i,t} \leq L_{\text{max},r} \quad r = 1, \ldots, R_b, i = 1, \ldots, I_r \\
& \quad L_{i,t} \geq L_{\text{min}} \quad i = 1, \ldots, I_b \\
& \quad L_{i,t} \leq L_{\text{max}} \quad i = 1, \ldots, I_b
\end{align*}
\]

The linear program seeks to:

- Minimize the workload of the most loaded toolset in the balancing group \( L_{\text{max}} \).
- Maximize the workload of the least loaded toolset in the balancing group \( L_{\text{min}} \).
- Minimize the total workload of toolsets \( \sum_i L_{i,t} \) and maximize the total workload of the least loaded toolset per recipe \( \sum_r L_{\text{min},r} \), with the same degree of priority.
- Minimize the total workload of the most loaded toolsets per recipe \( \sum_r L_{\text{max},r} \).

For the example cited above, the remaining steps of 10 lots are considered to be processed by 6 toolsets \{M1, M2, M3, M4, M5, M6\}. These toolsets are classified into 4 balancing-groups \{M1, M6\}, \{M2, M4\}, \{M3\} and \{M5\}. Figure 7 illustrates the saturation percentage of toolsets i.e. the ratio of the loading to the available capacity during the first period of the planning horizon \( \frac{L_{\text{max}}}{C_{\text{TCoeff}}}, i = 1..6 \) while processing the remaining steps ordered in increasing order of the start date.

In this example, the capacity of all the considered toolsets \( (C_{\text{TCoeff}}, i = 1..6) \) is equal to 24 hours/day. Figure 7 shows that there are two over-saturated toolsets (M2 and M6) which workloads exceed saturation threshold.

![Workload accumulation at infinite capacity for the first period of the planning horizon.](image)

**C. Workload/Capacity Balancing Module**

As a result of the workload accumulation module, loading of some toolsets may exceed their maximal capacities i.e. constraints (13) are not satisfied. In this case, the toolset is unable to process all its affected steps during the considered period so its loading should be balanced over subsequent periods. The principle of this module is to postpone additional lots in order to bring back workload of over-saturated toolsets below their maximal saturation and to smooth the activity over the planning horizon. The algorithm for workload/capacity balancing module is as follows:

1. Sort toolsets in decreasing order of saturation.
2. Select lots executed on over-saturated toolsets.
3. Sort selected lots in increasing order of a computed ranking coefficient \( \text{rankingCoeff} \).
4. The \( \text{rankingCoeff} \) illustrates the priority of the lot in terms of its position in the process sequence of the considered toolset and the urgency of delivery. The position of a lot in the process sequence of a toolset is determined by the processing date of its last remaining step treated by the considered toolset denoted \( s_{\text{TCoeff}} \).

The urgency of delivery is defined by the lot cycle time coefficient \( \text{CTCoeff} \). To compute the \( \text{rankingCoeff} \), the lot position in the process se-
Hence, this module modifies steps projection at period $t$ as well as the WIP for the beginning of period $t + 1$.

For instance, to balance the capacity and the workload of the considered toolsets $M2$ and $M6$ in the considered example, the balancing module selects $M2$ as the most over-saturated toolset ($\sum_{l,t}^a l = 109.3\%$). Then, it selects lots 2, 4, 5, 7, 8 and 10 processed by this resource (Figure 7). These lots are sorted in increasing order of $\text{rankingCoeff}_l$ as it is mentioned in Table V.

TABLE V: Order of lots processed on $M2$ according to $\text{rankingCoeff}_f$

<table>
<thead>
<tr>
<th>Lot $l$</th>
<th>$\text{CTCoeff}_l$</th>
<th>Step $s_l$</th>
<th>$s_{\text{S1},t}$</th>
<th>$\text{rankingCoeff}_f$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lot 5</td>
<td>1.07</td>
<td>Step 5.3</td>
<td>0.68</td>
<td>1.25</td>
</tr>
<tr>
<td>Lot 7</td>
<td>1.43</td>
<td>Step 7.2</td>
<td>0.265</td>
<td>1.43</td>
</tr>
<tr>
<td>Lot 10</td>
<td>0.79</td>
<td>Step 10.2</td>
<td>0.23</td>
<td>2.04</td>
</tr>
<tr>
<td>Lot 4</td>
<td>0.65</td>
<td>Step 4.2</td>
<td>0.2</td>
<td>2.34</td>
</tr>
<tr>
<td>Lot 8</td>
<td>0.48</td>
<td>Step 8.1</td>
<td>0</td>
<td>3.08</td>
</tr>
<tr>
<td>Lot 2</td>
<td>0.45</td>
<td>Step 2.1</td>
<td>0</td>
<td>3.22</td>
</tr>
</tbody>
</table>

In order to decrease the loading of the toolset $M2$, steps 5.3 and 7.2 and its successors are shifted to the next period of the planning horizon. Hence, the loading of $M2$ becomes less than its maximum capacity: $\sum_{l,t}^a l = 96.4\%$. $M4$ is also qualified for step 5.3, so its loading decreases by 5.06%. Step 5.4 projected in the first period is also postponed as it is the successor of the shifted step 5.3. Thus, the loading of $M1$ processing step 5.4 becomes equal to 58.83%. Shifting the successors of step 7.2 (steps 7.3, 7.4 and 7.5) leads to decreasing the loading of toolsets $M1$, $M4$ and $M5$. The same algorithm is applied to the toolset $M6$ by shifting step 9.1 and its successor step 9.2. So, its loading decreases to 72%.

The toolsets workload obtained after steps shifting is illustrated in Figure 8. Table VI presents the WIP and the computed parameters ($\text{RemPT}_t$, $\text{RemObjCT}_t$, $\text{RemExpCT}_t$ and $\text{CTCoeff}_f$) in the beginning of the next period.

The proposed approach is tested over a five-day planning horizon. Indeed, as mix variations were present in industrial dataset used for this study, it was decided to focus on a very short planning horizon to evaluate the proposed approach.

In order to decrease the loading of the toolset $M2$, steps 5.3 and 7.2 and its successors are shifted to the next period of the planning horizon. Hence, the loading of $M2$ becomes less than its maximum capacity: $\sum_{l,t}^a l = 96.4\%$. $M4$ is also qualified for step 5.3, so its loading decreases by 5.06%. Step 5.4 projected in the first period is also postponed as it is the successor of the shifted step 5.3. Thus, the loading of $M1$ processing step 5.4 becomes equal to 58.83%. Shifting the successors of step 7.2 (steps 7.3, 7.4 and 7.5) leads to decreasing the loading of toolsets $M1$, $M4$ and $M5$. The same algorithm is applied to the toolset $M6$ by shifting step 9.1 and its successor step 9.2. So, its loading decreases to 72%.

The toolsets workload obtained after steps shifting is illustrated in Figure 8. Table VI presents the WIP and the computed parameters ($\text{RemPT}_t$, $\text{RemObjCT}_t$, $\text{RemExpCT}_t$ and $\text{CTCoeff}_f$) in the beginning of the next period.

TABLE VI: WIP parameters in the beginning of the second period

<table>
<thead>
<tr>
<th>Lot $l$</th>
<th>$\text{CTCoeff}_f$</th>
<th>$\text{RemPT}_t$ in days</th>
<th>$\text{RemObjCT}_t$ in days</th>
<th>$\text{RemExpCT}_t$ in days</th>
<th>$\text{CTCoeff}_f$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lot 1</td>
<td>0.33</td>
<td>6</td>
<td>1.1</td>
<td>1.6</td>
<td>4</td>
</tr>
<tr>
<td>Lot 3</td>
<td>0.5</td>
<td>2</td>
<td>0.25</td>
<td>0.41</td>
<td>0.5</td>
</tr>
<tr>
<td>Lot 4</td>
<td>0.5</td>
<td>3</td>
<td>0.58</td>
<td>0.76</td>
<td>0.7</td>
</tr>
<tr>
<td>Lot 5</td>
<td>0.5</td>
<td>4</td>
<td>0.6</td>
<td>0.83</td>
<td>0.5</td>
</tr>
<tr>
<td>Lot 6</td>
<td>0.33</td>
<td>4</td>
<td>0.75</td>
<td>1.02</td>
<td>4</td>
</tr>
<tr>
<td>Lot 7</td>
<td>0.5</td>
<td>7</td>
<td>0.76</td>
<td>0.91</td>
<td>0.5</td>
</tr>
<tr>
<td>Lot 9</td>
<td>0.5</td>
<td>4</td>
<td>0.8</td>
<td>1.05</td>
<td>0.5</td>
</tr>
<tr>
<td>Lot 10</td>
<td>0.5</td>
<td>2</td>
<td>0.3</td>
<td>0.41</td>
<td>0.5</td>
</tr>
</tbody>
</table>

Clearly, the shorter the length of the period, the more accurate the results of the approach. The final obtained schedule for this instance is illustrated in Figure 9. For this instance, the TWT is equal to 1.46 days and we have five delayed lots.

Fig. 8: Workload accumulation at finite capacity.

Fig. 9: The obtained schedule using heuristic approach.

V. RESULTS AND DISCUSSION

The proposed algorithm is coded in Java and it is tested on a 4 GigaOctet RAM and 2.7 GigaHertz processor computer. We conducted two types of experiments to evaluate the performance of the proposed approach. The first type corresponds to a comparison between the exact method and the heuristic...
using a set of randomly generated instances. In the second type of experiment, we compare the projected schedule obtained by the proposed approach using real data with what is really going on in the wafer fab following this schedule.

A. Evaluation of the proposed heuristic algorithm in comparison with optimal solution

For this evaluation, random instances were generated and solved using the MIP and the proposed heuristic algorithm. The parameter, number of lots \( (L) \), assumes only five levels \( (L=10, 15, 20, 50 \) and \( L=100) \). The parameters generated for the proposed instances are presented in Table VII. So, three random problem instances for each fixed parameter combination are obtained, giving a total of 270 test problems. Each of the 270 problem instances generated has been solved using the ILOG CPLEX solver and the proposed heuristic algorithm.

<table>
<thead>
<tr>
<th>Problem Parameter</th>
<th>Values used</th>
<th>Total values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of lots ( (L) )</td>
<td>10, 15, 20, 50, 100</td>
<td>5</td>
</tr>
<tr>
<td>Maximum number of remaining steps of lot ( I ) (( \max S_I ))</td>
<td>10, 20, 30, 40, 50, 100</td>
<td>6</td>
</tr>
<tr>
<td>Number of toolsets ( I )</td>
<td>5, 10, 20</td>
<td>3</td>
</tr>
<tr>
<td>Number of time buckets ( T )</td>
<td>24</td>
<td>1</td>
</tr>
<tr>
<td>Weight per lot ( w_t )</td>
<td>uniform ((0,1))</td>
<td>1</td>
</tr>
<tr>
<td>Lots release dates ( r_t )</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Lots due dates ( d_t )</td>
<td>uniform ((1,30))</td>
<td>1</td>
</tr>
<tr>
<td>Lots quantity of wafers ( Q_I )</td>
<td>25</td>
<td>1</td>
</tr>
<tr>
<td>Steps unit process times ( p_{s,t,I} )</td>
<td>(0.0001 \times \text{uniform}(5,50))</td>
<td>1</td>
</tr>
<tr>
<td>Total parameter combinations</td>
<td>90</td>
<td></td>
</tr>
<tr>
<td>Total number of problem instances</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Total problems</td>
<td>270</td>
<td></td>
</tr>
</tbody>
</table>

The results on TWT obtained for each instance using MIP model and using the proposed iterative algorithm are recorded. Based on these results, the heuristic solution matched exactly with the optimal solution 53 times. Furthermore, for each instance with a size \( L \times \max S_I \times I \), we compute:

- The absolute deviation \( = |\text{TWT value from a heuristic algorithm - optimal TWT value}| \)
- The relative deviation \( = \frac{\text{absolute deviation value}}{\text{optimal TWT value}} \)

Figure 10 shows the relative deviation over 270 instances plotted against the absolute deviation.

In this figure, we can define four zones or classes according to the size of the instance:

- The first zone (corresponding to absolute deviation values \( \in [0..30] \) days and relative deviation values \( \leq 1 \)): Around 92% of the tested instances are situated in this zone. Hence, for most of the instances, the heuristic solution is close to the optimal one.

- The second zone (corresponding to absolute deviation values \( \in [30..140] \) days and relative deviation values \( \leq 1 \)): The 8 instances (\( \approx 3\% \) of the total of tested instances) belonging to this category are instances of large size \( (\geq 10000) \). For example, we find the instance with a size equal to 10000 \((L=100, \max S_I=10, I=10)\) which has an absolute deviation equal to 79 days and a relative deviation equal to 0.36. This instance has an optimal solution TWT equal to 218 days. The important value of the absolute deviation is thus not significant because of high values of TWT.

- The third zone (corresponding to absolute deviation values \( \in [0..30] \) days and relative deviation values \( > 1 \)): 14 instances (\( \approx 5\% \) of the total of tested instances) are located in this zone. We can cite the example of the instance with a size equal to 15000 \((L=50, \max S_I=30, I=10)\), a low value of absolute deviation equal to 2.23 days and a high value of relative deviation equal to 4.74. For this instance, both of the optimal and the approximate solutions present a low value of TWT. Hence, in this zone, the importance of the relative deviation has no significance.

- The fourth zone (corresponding to absolute deviation values \( > 30 \) days and relative deviation values \( > 1 \)): No instance is located in this zone characterized by high values of absolute and relative deviations.

B. Experimental tests on real fab data

The aim of this section is to evaluate the ability of the proposed approach to tackle real world problems. The test of the real instance \((L=2000, \max S_I=680, I=300, T=24)\), unsolved in reasonable execution time using the MIP approach in Section 3, is treated. The execution time of this instance with the proposed algorithm is around 30 seconds. In the calculated production schedule, 80% of projected lots are delivered on time. Furthermore, the saturation of toolsets is kept below the predefined saturation threshold while minimizing lots lateness. Figure 11 illustrates the obtained weekly saturation at infinite and finite capacity of a photo-lithography toolset considered as a bottleneck. In semiconductor fabs, several indicators are used to measure performance [52]. Jointly with
managers of the fab, we identified three relevant indicators for our study, as described below:

- **Number of moves**: This corresponds to the number of completed steps on each period of the planning horizon, which can be compared to the real number in the production line.

- **Number of moves by usage**: It is the number of processed steps by set of toolsets belonging to the same area named "usage" in each period of the planning horizon.

- **Total Weighted Tardiness TWT**: This indicator is used to evaluate the waiting times of lots for processing.

In this section, we compare the cited indicators of performance of the heuristic solution with the indicators determined in the real production line. To ensure this experiment, six tests have been performed on actual instances issued over four months of production: September, October, November and December 2015. We have made projections in six different periods (week1, week2, week3, week4, week5 and week6) and we have determined the three indicators for each projection. For confidential reasons, we are not allowed to provide the real values of the fab. This is why, we compute the relative deviation between the predicted value and the real one for each period of the planning horizon:

\[
\text{Relative deviation} = \frac{|\text{Estimated value} - \text{real value}|}{\text{real value}}
\]

1) **Analysis based on the performance measure: number of moves**: Figure 12 shows relative deviations of number of moves over 15 time buckets (weeks) of the planning horizon. It illustrates that in the first 6 periods for the different instances, the relative deviation between the real number of processing steps and the calculated value is low. The average of the average relative deviations over six periods for the different tests is equal to 12.7%, reflecting a small difference between the estimated number of moves and the achieved one. Further being away from the beginning of the projection, the relative deviation between the obtained solution and the real number of moves increases which is explained by the variability of the process. Hence, there is a convergence between what is estimated and what is achieved in terms of periodic activity for a short-term planning horizon.

2) **Analysis based on the performance measure: number of moves by usage**: To evaluate how the heuristic solution anticipates the fab loading, we compute the absolute deviation of the number of moves by set of toolsets sharing the same qualifications named "usage" over the six instances for each period of the planning horizon. Figure 13 shows the difference between the total number of completed steps processed by two types of bottleneck usages (photolithography and etching). For this indicator also, we observe a convergence between the planning and the real process for the first 6 periods with an average of the average relative deviations over these periods equal to 6.5% for the usage of photolithography and 12.3% for the usage of etching. Therefore, the heuristic provides good estimations of the tools loading close to the real workload while respecting capacity constraints.

3) **Analysis based on the performance measure: TWT**: To compare between the real total weighted tardiness and the obtained value of this indicator using the iterative algorithm for the six tests, absolute and relative deviations are computed and reported in Table VIII. From Table VIII, we note that the estimated value of TWT is close to the real tardiness while respecting lots due dates. Indeed, the average of relative deviations over six instances is equal to 4%.

**VI. CONCLUSIONS AND PERSPECTIVES**

This paper has examined the problem of WIP projection at finite capacity to minimize the TWT, and has proven empirically the computational complexity in obtaining optimal solution and suggested a simple, fast and efficient heuristic.
The motivation for this research is to compute a feasible production plan to drive the execution of wafer fabs. This problem is of considerable practical value because the heuristic, proposed in this paper, can be used in planning of a large number of production lots while respecting lots due dates and capacity and capability constraints.

The computational tests, made on real production instances, showed that acceptable solutions are obtained in reasonable execution time. Indeed, the TWT could be minimized and the average tool utilization rate could be balanced significantly by using the developed system. Besides, the computation for real instances is achieved in around 30 seconds which is efficient for planning problems with a horizon of weeks up to months in real situations. Hence, this decision support tool outperforms simulation and analytic models for establishing a feasible production schedule rapidly. Finally, it is observed (as well as statistically verified) from the results of the comparison of the different criteria (total number of moves, number of moves by usage and TWT) an obvious convergence between what is predicted using the developed approach and what is achieved in the real process over a short-term planning horizon. These results show that the implementation of the finite capacity planning system in real fabs seems very interesting to minimize lots lateness and to establish a feasible production schedule. There are a number of interesting extensions of the problems that can be pursued. The first important issue would be to perform a more thorough multi-criteria analysis while shifting lots to balance toolsets loadings. Besides, it is necessary to implement the developed finite capacity planning system in the production plant to guarantee the performance of the solution. Considering other specificities of semiconductor industry such as batching or sequence dependent setup times may be interesting to enhance the accuracy of the developed system.

ACKNOWLEDGMENT

This work is supported by the ENIAC European Project INTEGRATE. The authors also gratefully acknowledge STMicroelectronics for their support on the knowledge of the semiconductor industry.

REFERENCES


### Table VIII: TWT comparison actual versus forecast

<table>
<thead>
<tr>
<th>Instance</th>
<th>Absolute deviation (days)</th>
<th>Relative deviation (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instance 1</td>
<td>228.255</td>
<td>6</td>
</tr>
<tr>
<td>Instance 2</td>
<td>98.305</td>
<td>2.62</td>
</tr>
<tr>
<td>Instance 3</td>
<td>108.86</td>
<td>3.39</td>
</tr>
<tr>
<td>Instance 4</td>
<td>47.77</td>
<td>1.83</td>
</tr>
<tr>
<td>Instance 5</td>
<td>50.13</td>
<td>2.08</td>
</tr>
<tr>
<td>Instance 6</td>
<td>146.23</td>
<td>7.92</td>
</tr>
</tbody>
</table>