Optimizing Power Peaks in Simple Assembly Line Balancing through Maximum Satisfiability

1st Zhifei Zheng
School of Software
Yunnan University
Kunming, China
zzfreya@mail.ynu.edu.cn

2nd Sami Cherif
Laboratoire MIS UR 4290
Université de Picardie Jules Verne
Amiens, France
sami.cherif@u-picardie.fr

3rd Rui Sá Shibasaki
Laboratoire MIS UR 4290
Université de Picardie Jules Verne
Amiens, France
rui.sa.shibasaki@u-picardie.fr

2024 IEEE 36th International Conference on Tools with Artificial Intelligence (ICTAI) | 979-8-3315-2723-5/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICTAI62512.2024.00060

***Abstract*—The Simple Assembly Line Balancing Problem with Power Peak Minimization (SALB3PM) is a relatively new problem that aims to assign tasks to workstations with a focus on minimizing power peaks. By integrating load balancing and task scheduling, this problem offers a comprehensive approach to enhancing energy efficiency in production systems, which can lead to significant cost savings alongside a positive environmental impact. This paper introduces novel models for SALB3PM based on Maximum Satisfiability (MaxSAT), the natural optimization extension of the Satisfiability problem, providing a new perspective to solve this optimization problem effectively. Experimental results demonstrate the efficiency and robustness of our approach with respect to the MaxSAT solvers applied. To the best of our knowledge, this is the first attempt to address the SALB3PM problem through the lens of Maximum Satisfiability.**

***Index Terms*—SALB3PM, Maximum Satisfiability, Modeling**

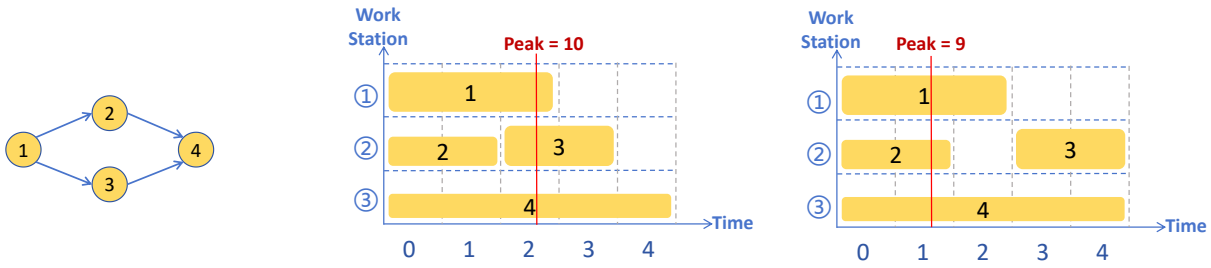
I. Introduction

Simple assembly lines are streamlined manufacturing systems comprising a series of workstations arranged in sequence and typically linked by a conveyor belt [1]. These workstations operate simultaneously and must sequentially process a set of tasks. To maintain a certain production pace, all tasks assigned to a workstation must be completed within a specified time interval, referred to as the cycle time. Each task must be entirely handled by a unique workstation. Moreover, precedence constraints between tasks must be satisfied. In this context, the simple assembly line balancing problem (SALBP) consists in assigning tasks to workstations while optimizing a specific objective [2]. There are several variants of SALBP with the corresponding methodologies proposed in the literature review. For instance, SALBP-1 minimizes the number of workstations for a given cycle time [3], [4], [5], while SALBP-2 minimizes the cycle time for a given number of workstations [6], [7], [8]. On the other hand, SALBP-E minimizes the product of cycle time and the number of workstations [9], [10] whereas SALBP-F is a well-known decision variant which simply determines the feasibility of a solution [2].

The present work deals with a relatively new variant of SALBP: the Simple Assembly Line Balancing Problem with Power Peak Minimization (SALB3PM). First introduced in [11], SALB3PM consists in finding a feasible assignment of tasks to workstations that minimize the overall peak of power consumption under the given fixed number of workstations and the fixed cycle time. This problem is particularly relevant in the context of energy-efficient manufacturing, where reducing peak power consumption can lead to significant cost savings and environmental benefits. An integer linear programming (ILP) model for SALB3PM was proposed in [11], and a metaheuristic was presented in [12]. In the latter work, the authors considered a specialized SALB3PM variant where idle times between tasks are not allowed. The preliminary study in [13] introduces a new approach based on satisfiability to solve SALB3PM, with promising results. Yet, this approach applies a simple iterative process to optimize the solution cost and therefore does not harness the power of recent solvers dedicated to optimization in the context of satisfiability. More specifically, Maximum Satisfiability (MaxSAT) is the natural optimization extension of the well-known Satisfiability (SAT) problem, where the goal shifts to computing the maximum number of clausal constraints that can be satisfied by an assignment of the variables in a given propositional formula [14], [15]. Although MaxSAT is NP-hard, the last decade has known major breakthroughs in MaxSAT theory and solving, making it a powerful formalism that can be used to model and effectively solve many academic and industrial problems, particularly in the domain of planning and scheduling [16], [17], [18], [19], [20], [21].

In this paper, we aim to harness the strengths of MaxSAT in handling combinatorial optimization problems and particularly SALBP3PM. Building upon the model of Gianessi et al. in [11] and the subsequent contributions of Py et al. in [13], we introduce a new SALB3PM model based on MaxSAT. To the best of our knowledge, this is the first attempt to address the SALB3PM problem through the lens of Maximum Satisfiability. Furthermore, we enhance our formulation through a binary representation, enabling the model to reduce the number of variables and the potential search space. We also conduct a thorough experimental evaluation demonstrating the efficiency and robustness of our approach. This paper is organized as follows. Section II presents necessary definitions and notations and reflects on two important related works proposed in the previous literature. Our dedicated MaxSAT models are introduced in Section III and evaluated in Section IV. Finally, we conclude and discuss future work in Section V.

2375-0197/24/$31.00 ©2024 IEEE
DOI 10.1109/ICTAI62512.2024.00060



(a) Precedence Graph (b) Candidate Assignment (c) Optimal Assignment

Fig. 1: An SALB3PM instance with $n = 4$, $m = 3$ and $c = 5$

II. Preliminaries

A. SALBP3PM

As strengthened in [11], SALB3PM is the first variant of SALBP that considers energy peaks at a strategic level. Besides the basic constraints in the other variants of SALBP, tasks in SALB3PM feature an extra power consumption parameter. Therefore the objective alters to minimize the power peak over time, which could lead to energy savings and lower carbon emissions. By integrating load balancing and task scheduling, SALB3PM provides a more comprehensive approach to optimizing energy efficiency in production systems and beyond.

SALB3PM is defined as follows. Let N and S be, respectively, a set of n tasks and m workstations. Each task $i \in N$ has an execution time $t_i \in \mathbb{N}^*$ and a power consumption $w_i \in \mathbb{N}^*$. We denote c the cycle time, divided into time units denoted $T = \{0, ..., c-1\}$. $T^i = \{0, ..., c - t_i\}$ indicates the set of the feasible start times for task $i \in N$. We denote $G = (N, P)$ the directed graph of precedence relations between tasks, where P is the set of arcs $(i, j) \in N \times N$, representing the precedence relation $i \prec j$. Hence, for an arc (i, j), either task i is allocated to a workstation with a smaller ID than task j, or they are assigned to the same workstation with i scheduled to start earlier than j. SALB3PM thus consists in assigning tasks to workstations so as to minimize peak power consumption while meeting precedence constraints and cycle time.

It is worth noting that, in SALB3PM, the specific time unit at which the power peak occurs depends on the order in which tasks are processed on each workstation. This is in contrast to other variants of SALBP, where the task sequence within a workstation does not affect the overall solution. Figure 1 illustrates two feasible assignments for an SALB3PM instance with $n = 4$, $m = 3$, $c = 5$, $(t_i)_{1 \leq i \leq 4} = (3, 2, 2, 5)$, $(w_i)_{1 \leq i \leq 4} = (4, 3, 4, 2)$, and $P = \{(1, 2), (1, 3), (2, 4), (3, 4)\}$. The first assignment (Figure 1b) leads to the peak power consumption of 10, whereas the second assignment (Figure 1c) results in a peak power consumption of 9, which is the optimal solution for this instance.

B. Maximum Satisfiability

Let V be the set of propositional variables. A literal l is a variable $x \in V$ or its negation $\overline{x}$. A clause is a disjunction of literals. A formula in Conjunctive Normal Form (CNF) is a conjunction of clauses. An assignment $\alpha : V \to \{true, false\}$ maps each variable to a Boolean value and can be represented as a set of literals. A literal l is satisfied by an assignment α if $l \in \alpha$, else it is falsified by α. A clause is satisfied by an assignment α if at least one of its literals is satisfied by α, otherwise it is falsified by α. A CNF formula is satisfiable if there exists an assignment α which satisfies all its clauses, else it is unsatisfiable. The cost of an assignment α, denoted $cost_\alpha(\phi)$, is the number of clauses in the formula ϕ falsified by α. Solving the Satisfiability (SAT) problem consists in determining whether a given CNF formula ϕ is satisfiable whereas, for Maximum Satisfiability (plain MaxSAT) [14], [15], the goal shifts to determining the maximum number of clauses that can be satisfied in ϕ by an assignment of the variables or, equivalently, the minimum number of clauses that each assignment must falsify, i.e., $optimum(\phi) = \min_\alpha cost_\alpha(\phi)$.

The partial MaxSAT problem is a well-known variant taking as input a partial CNF formula, i.e., a bipartite set of clauses $\phi = H \cup S$ where H is the set of hard clauses that must be satisfied and S is the set of soft clauses to be optimized as in plain MaxSAT. The goal thus shifts to computing $opt(S)$ such that all the clauses in H are satisfied. Another well-known variant is weighted MaxSAT which takes as input a weighted CNF formula ϕ, i.e., a set of tuples (C, w_C) where C is a clause and $w_C \in \mathbb{N}^*$ is its associated weight (hard clauses are typically associated with infinite weight), and seeks $optimum(\phi)$ where $cost_\alpha(\phi)$ now denotes the sum of weights of clauses in ϕ falsified by α. Finally, we mention that previous literature introduces Pseudo-Boolean (PB) constraints which can be efficiently encoded in CNF form [22], [23]. These constraints will be used in our subsequent models and are of the following form:

$$\sum_j a_j * l_j \rhd b \qquad \text{where } a_j \in \mathbb{N}, b \in \mathbb{N}, l_j \text{ is a literal and } \rhd \in \{=, >, \geq, <, \leq\}.$$

C. Related Work

Two important models for SALB3PM exist in the literature review: the ILP-based model proposed in [11] and the SAT-based model proposed in [13]. In [11], the SALB3PM is modeled into the ILP model showcased in Figure 2, with the following decision variables:

- **Binary assignment variables** $X_{i,s}$ with $i \in N$ and $s \in S$, which are set to 1 if task i is assigned to the station s, 0 otherwise.
- **Binary trigger variables** $B_{i,t}$ with $i \in N$ and $t \in T$, which are set to 1 when task i starts at time unit t, 0 otherwise.
- **Integer power peak variables** W_{max} which represent an upper bound on the power consumption peak.

The constraints of this model can be divided into two categories: those inherited from the classical SALBP (Constraints ILP-2 - ILP-4) and those specific to SALB3PM (Constraints ILP-1, ILP-5 - ILP-8). In the first category, Constraint ILP-2 ensures that every task j is allocated to a single workstation k, while Constraint ILP-3 limits the total processing time of tasks on each workstation to the given cycle time c. Moreover, Constraint ILP-4 guarantees that precedence relations are respected by assigning each predecessor task i to the same or an earlier workstation as its successor j. The SALB3PM-specific constraints, on the other hand, deal with task scheduling and power consumption aspects. Constraint ILP-5 makes sure that each task j is processed exactly once, starting at some time unit t within its feasible time window T^j. Constraint ILP-6 works in conjunction with Constraint ILP-4 to enforce precedence: if $i \prec j$ and both tasks are allocated to the same workstation, then i must start at least t_i time units before j; otherwise, this constraint becomes redundant. Constraint ILP-7 prevents any two tasks from being processed simultaneously on the same workstation. Lastly, Constraint ILP-8 imposes an upper bound W_{max} on the total power consumption of all running tasks at any time unit t. Combined with the objective function ILP-1 that minimizes W_{max}.

$$\min W_{max} \tag{ILP-1}$$

s.t.

$$\sum_{k \in S} X_{j,k} = 1 \qquad \forall j \in N \tag{ILP-2}$$

$$\sum_{j \in N} t_j \cdot X_{j,k} \leq c \qquad \forall k \in S \tag{ILP-3}$$

$$X_{j,k} \leq \sum_{h \in S : h \leq k} X_{i,h} \qquad \forall i,j,k \in N^2 \times S \text{ s.t. } i \prec j \tag{ILP-4}$$

$$\sum_{t \in T^j} B_{j,t} = 1 \qquad \forall j \in N \tag{ILP-5}$$

$$B_{j,t} \leq \sum_{\tau=0}^{t-t_i} B_{i,\tau} + 2 - X_{i,k} - X_{j,k} \qquad \forall i,j,k,t \in N^2 \times S \times T^j \text{ s.t. } i \prec j \tag{ILP-6}$$

$$X_{i,k} + X_{j,k} + \sum_{\tau=t-t_i+1}^{t} B_{i,\tau} + \sum_{\tau=t-t_j+1}^{t} B_{j,\tau} \leq 3 \qquad \forall i,j,k,t \in N^2 \times S \times T \tag{ILP-7}$$

$$\sum_{j \in N} w_j \cdot \left(\sum_{\tau=t-t_j+1}^{t} B_{j,\tau} \right) \leq W_{max} \qquad \forall t \in T \tag{ILP-8}$$

$$X_{i,k}, B_{i,t} \in \{0,1\}, W_{max} \in \mathbb{Z}_+ \tag{ILP-9}$$

Fig. 2: The ILP-based model for SALB3PM proposed in [11]

The SAT-based model proposed in [13] is built upon the above model. Precisely, it features the following additionnal aspects. A new series of decision variables, which we can refer to as activity variables $A_{i,t}$, are introduced to simplify some constraints. The algorithm first models the feasibility problem as a propositional formula in CNF form, without considering the energy aspects. This CNF formula is then fed to a SAT oracle to check its satisfiability. If a solution is found, the algorithm gradually adds new constraints to the formula to account for the energy impacts of the decisions made. This iterative process aims to obtain better quality solutions in terms of energy consumption, while ensuring the satisfiability of the updated formula at each step.

III. MaxSAT Models for SALB3PM

A. Inaugural Model

An SALB3PM instance can be naturally transformed into a MaxSAT instance as follows. To find the optimal peak value W_{peak}, we establish a lower bound LB and an upper bound UB to narrow down the range. More specifically, since all the tasks will be assigned to the workstations, the maximum power consumption among all the tasks is a lower bound on W_{peak}. On the other hand, since at most m tasks can be active at the same time on different stations, we can simply compute an upper bound by summing the m highest power consumption values among all the tasks. Formally, we have :

$$LB = \max_{i \in \{1,2,\ldots,n\}} w_i \quad and \quad UB = \sum_{w_i \in W_m} w_i$$

where W_m is the set of the m highest values of w_i. This approach allows us to search for the optimal solution more efficiently by reducing the search space. We define the following Boolean variables for our model:

- **Assignment variables** $X_{i,s}$ with $i \in N$ and $s \in S$, which are set to 1 if task i is assigned to the station s, 0 otherwise.
- **Trigger variables** $B_{i,t}$ with $i \in N$ and $t \in T$, which are set to 1 when task i starts at time unit t, 0 otherwise.
- **Activity variables** $A_{i,t}$ with $i \in N$ and $t \in T$, which are set to 1 when task i is being executed at time unit t, 0 otherwise.
- **Solution variables** U_j with $j \in \{1, ..., UB\}$, which are set to 1 when the obtained peak power consumption is larger than or equal to j, 0 otherwise.

Figure 3 showcases our proposed inaugural MaxSAT model for the SALB3PM problem with Constraints 1 to 11. The model minimizes peak power consumption using soft clauses in Constraint 1 involving literals $\overline{U_j}$ for each power consumption unit. More specifically, we have:

$$W_{peak} = \sum_{j \in \{1,\ldots,UB\}} U_j$$

Hard clauses enforce the precedence and cycle time constraints while maintaining the problem's logical structure. Constraint

Fig. 3: The inaugural MaxSAT model for SALB3PM

Soft Constraints:

$$\overline{U_j} \qquad \forall j \in \{1, ..., UB\} \tag{1}$$

Hard Constraints:

$$\sum_{s \in S} X_{i,s} = 1 \qquad \forall i \in N \tag{2}$$

$$\overline{X_{i,s_1}} \vee \overline{X_{j,s_2}} \qquad \forall (i, j, s_1, s_2) \in N^2 \times S^2 \text{ s.t. } i \prec j \text{ and } s_1 > s_2 \tag{3}$$

$$\overline{X_{i,s}} \vee \overline{X_{j,s}} \vee \overline{B_{i,t_1}} \vee \overline{B_{j,t_2}} \qquad \forall (i, j, s, t_1, t_2) \in N^2 \times S \times T^2 \text{ s.t. } i \prec j \text{ and } t_1 > t_2 \tag{4}$$

$$\sum_{t \in T} B_{i,t} = 1 \qquad \forall i \in N \tag{5}$$

$$\overline{B_{i,t}} \qquad \forall (i, t) \in N \times (T \setminus T^i) \tag{6}$$

$$\overline{B_{i,t}} \vee A_{i,t+\epsilon} \qquad \forall (i, t, \epsilon) \in N \times T^i \times \{0, ..., t_i - 1\} \tag{7}$$

$$\overline{X_{i,s}} \vee \overline{X_{j,s}} \vee \overline{A_{i,t}} \vee \overline{A_{j,t}} \qquad \forall (i, j, s, t) \in N^2 \times S \times T \text{ s.t } i < j \tag{8}$$

$$U_j \qquad \forall j \in \{1, ..., LB\} \tag{9}$$

$$\overline{U_j} \vee U_{j-1} \qquad \forall j \in \{2, ..., UB\} \tag{10}$$

$$\sum_{i \in \{1, ..., n\}} w_i * A_{i,t} + \sum_{j \in \{1, ..., UB\}} \overline{U_j} \leq UB \qquad \forall t \in T \tag{11}$$

2 ensures that each task is assigned to exactly one station. Constraint 3 and 4 satisfy the precedence relations, ensuring that tasks are performed in the correct order across the stations. Constraint 5 ensures that each task must start at exactly one feasible time unit. Constraint 6 enforces that tasks must start within their feasible time windows. Constraint 7 ensures proper task activation and Constraint 8 is used to prevent simultaneous execution of tasks on the same station. Constraints 9, 10 and 11 are related to power peaks. In particular, Constraint 9 enforces the computed lower bound. Constraint 10, i.e. $U_j \rightarrow U_{j-1}$, maintains the order of power unit usage. Finally, Constraint 11 encapsulates the overall power consumption constraint, ensuring it remains within the obtained W_{peak} across all time units.

B. Model With Binary Representation

In MaxSAT problems, the potential search space is bounded by the number of Boolean variables. Since UB can have high values, we propose to introduce a model based on a binary representation to save on the number of Boolean variables. More specifically, we replace the solution variables presented in Section III-A by binary-represented ones $binU_j$ with $j \in \{0, ..., \lceil \log_2 UB \rceil\}$. As such, $binU_j$ takes the value of 1 when the j-th bit of the obtained solution is 1 and 0 otherwise. Note that, in this case, we have:

$$W_{peak} = \sum_{j \in \{0, ..., \lceil \log_2 UB \rceil\}} 2^j * binU_j.$$

For example, if the solution is $6 = (110)_2$, then the binary-encoded solution variables would be $binU_2 = 1$, $binU_1 = 1$, and $binU_0 = 0$. The binary representation can be integrated into our previous model by maintaining Constraints 2 to 8 and implementing specific modifications for the other constraints as showcased in Figure 4. In particular, Constraint 1' aims to minimize the peak power consumption using a binary representation, where each binary unit $binU_j$ is weighted by 2^j. This approach allows for a more efficient representation of the power peak values. The inaugural model's hard clauses (equation 2 to 8) are retained to enforce the fundamental constraints of task assignments, precedence, and time scheduling. Constraint 9' is an adaption of equation 9, which ensures that the final obtained power peak value, represented in a

Fig. 4: Enhanced MaxSAT model with binary representation for SALB3PM

Soft Constraints:

$$(\,\overline{binU_j}\,,\,2^j\,) \qquad \forall j \in \{0,\,...,\,\lceil \log_2 UB \rceil\} \tag{1'}$$

Hard Constraints:

$$\text{Constraints 2 to 8 from previous model} \tag{2'-8'}$$

$$\sum_{j \in \{0,\,...,\,\lceil \log_2 UB \rceil\}} 2^j * binU_j \geq LB \qquad \forall t \in T \tag{9'}$$

$$\sum_{i \in \{1,2,...,n\}} w_i * A_{i,t} + \sum_{j \in \{0,1,...,\lceil \log_2 UB \rceil\}} 2^j * \overline{binU_j} \leq UB' \qquad \forall t \in T \tag{11'}$$

binary format, must be larger than or equal to the lower bound (LB). Constraint 11' is an adaption of equation 11, using binary variables and ensuring that the total power used at any moment is within the limits of the calculated power peak, thus providing precise control over the power usage. Note that, in this last constraint, we set the following value for the right-hand side:

$$UB' = \sum_{j \in \{0,\,...,\,\lceil \log_2 UB \rceil\}} 2^j$$

Together, the new Constraints (1' to 11') form our enhanced model, offering a more compact solution space compared to the inaugural formulation.

IV. Experimental Evaluation

A. Experimental Protocol

The experiments were carried out on an H620-G30 server, which runs on CentOS Linux release 7.6.1810 (Core) and is equipped with two Hygon C86 7185 32-core processors that can be clocked up to 2GHz. For our experiments, we use instances adapted from a SALBP-1 dataset[1]. This dataset contains the well-known test sets of [24] and [25], which have been widely used for testing and comparing solution procedures in almost all relevant studies since the early nineties [26]. The dataset includes various precedence graphs and detailed descriptions of the file format. In our experiment, the precedence constraints and the cycle time c are given in the original dataset, while the number of stations m is set as the optimal value m^* obtained in SALBP-1, ensuring the feasibility of every generated SALB3PM instance. The power consumption values w_i are generated from the uniform distribution $U(5, 50)$. The same approach has also been used in [11] and [13], with 23 and 21 generated instances, respectively. Nevertheless, our dataset contains a total of 97 instances with varying n, m, c values, and precedence graphs, providing a more comprehensive testbed for the SALB3PM problem.

[1]https://assembly-line-balancing.de/salbp/benchmark-data-sets-1993/

We employed the following state-of-the-art solvers: CPLEX [27] for the ILP model proposed in [11], Cadical [28] for the SAT model proposed in [13], and the three MaxSAT solvers MaxHS [29], EvalMaxSAT [30], and WMaxCDCL [31], [32] for solving our MaxSAT models, both the inaugural version (*inaugural*) and the version with binary representation (*Bin.*). It is important to note that the three selected MaxSAT solvers are respectively ILP-based, SAT-based and Branch&Bound-based MaxSAT solvers with the latter two being first-placed solvers in the last edition of the MaxSAT evaluation[2]. All solvers were used with their default settings, allowing for a reproducible evaluation of our proposed models, although it is worth noting that MaxSAT solvers offer various configuration options that can be tuned for specific problem instances. To conduct the SAT and MaxSAT encoding, we used the PySAT[3] library [33], with PB constraints encoded in the Binary Merger mode proposed in [34] and provided by the PBEnc class of PySAT. We recall that this encoding enables to write PB constraints in CNF form using only $\mathcal{O}(h^2 \log^2(h) \log(a_{\mathsf{max}}))$ newly introduced variables and clauses, where h is the number of literals in the weighted sum and a_{max} the maximal weight. We have set the cutoff time to 3600s for each instance and we have conducted experiments to evaluate the efficiency of the proposed inaugural MaxSAT model and the enhanced formulation with the binary representation, whose results are reported hereafter.

B. The Overall Performance

Figure 5 represents each solver's runtime as a function of the cumulative optimally solved instances. We can observe that the inaugural MaxSAT model run by the three solvers and the SAT model run by Cadical have comparable performances in general. To be more specific, within the cutoff time, the number of solved instances using the inaugural model corresponds to 48, 48, and 47 respectively with EvalMaxSAT, MaxHS, and WMaxCDCL; while the SAT-based method using Cadical manages to solve 46 instances. For the model enhanced with

[2]https://maxsat-evaluations.github.io/2023/index.html
[3]https://pysathq.github.io/

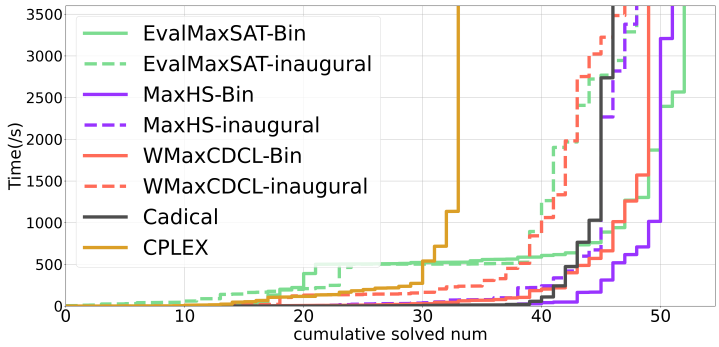


Fig. 5: Cumulative solved number of instances with respect to solving time in seconds for the ILP model run by CPLEX, the SAT model run by Cadical, and the two MaxSAT models run by EvalMaxSAT, MaxHS, and WMaxCDCL. "-inaugural" refers to the inaugural model proposed in III-A, which is represented by dashed lines in the plot, and "-Bin" refers to the model with binary representation proposed in III-B, which is represented by solid lines.

Family	#I	ILP	SAT	MaxSAT (inaugural)			MaxSAT (binary)		
		CPLEX	Cadical	WMaxCDCL	EvalMaxSAT	MaxHS	WMaxCDCL	EvalMaxSAT	MaxHS
BUXEY	7	1 (272)	3 (3780)	4 (7974)	2 (2916)	3 (3679)	3 (845)	**4 (4306)**	3 (190)
GUNTHER	7	0 (0)	4 (47)	4 (3731)	5 (4051)	4 (322)	5 (1799)	5 (3553)	**5 (1046)**
LUTZ2	11	0 (0)	2 (130)	2 (781)	2 (3869)	2 (640)	2 (404)	2 (1373)	**2 (90)**
SAWYER30	9	2 (266)	3 (34)	3 (3024)	4 (5635)	4 (2454)	4 (1766)	5 (3846)	**5 (1170)**
WARNECKE	16	0 (0)	1 (765)	1 (1980)	1 (2768)	1 (2818)	1 (1013)	1 (1871)	**1 (710)**
Total	50	3 (539)	13 (4756)	14 (17490)	14 (19240)	14 (9913)	15 (5827)	**17 (14949)**	16 (3206)

TABLE I: Number of optimally solved instances for major families within the benchmark. For each family, we list the number of instances ($\#I$) and the performances of each solver, using the format $num.(time.)$, where $num.$ indicates the number of optimally solved instances and $time.$ indicates the corresponding accumulated time in seconds. For each family, the largest solved numbers with the shortest run-time are marked in bold.

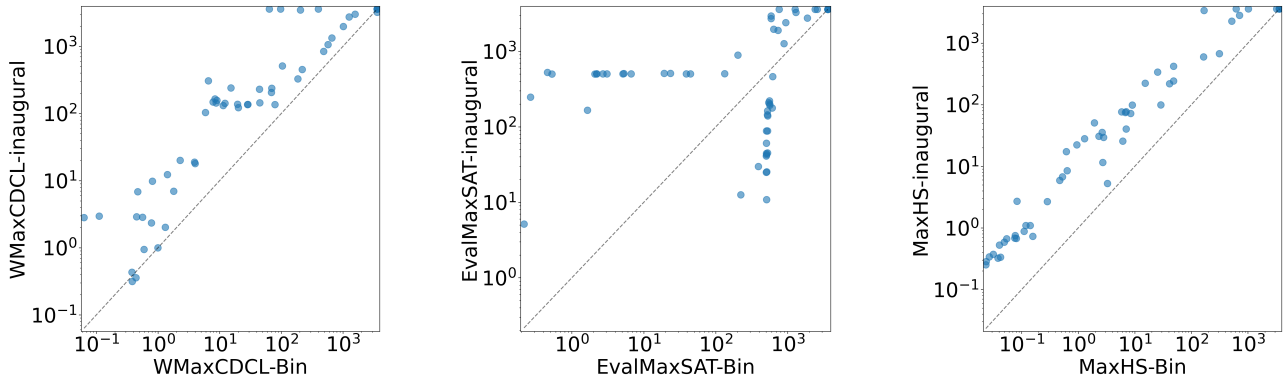
the binary representation, MaxHS exhibits remarkable stability and consistency throughout the whole benchmark, maintaining a steady lead over other solvers, with 51 instances solved. EvalMaxSAT, while trailing MaxHS in the early stages, manages to close the gap and eventually surpasses all other solvers, solving the highest number of instances within the time limit (resp. 52). WMaxCDCL (resp. 49) and Cadical (resp. 46) place third and fourth in terms of solved instances. Notably, the original ILP model, although tested with the powerful general-purpose optimizer CPLEX, seems to acheive the worst performance as it solves only 33 instances. This underscores the relevance of applying satisfiability-based methods, particularly our MaxSAT-based approach in the context of SALB3PM. Indeed, the results clearly demonstrate the superior performance of our MaxSAT-based method with the binary representation. The improved performance can be attributed to several factors, mainly the effectiveness of state-of-the-art MaxSAT solvers in handling complex constraint optimization problems as well as our SALB3PM tailored-modeling approach and, particularly, the binary representation's ability to reduce the problem size and improve solver efficiency.

Table I displays the detailed results for the five biggest instance families: BUXEY, GUNTHER, LUTZ2, SAWYER30, and WARNECKE. These families contain instances sharing the same precedence relations but vary in the number of workstations and cycle times. We can observe that for all the families, the MaxSAT models consistently solve more instances to optimality with smaller runtime compared to the ILP and SAT models. On our MaxSAT model with the binary representation, EvalMaxSAT manages to solve the most number of instances whereas MaxHS ranks second with remarkable speed.

Furthermore, in table II, we report the running times for hard individual instances whose minimum running times across all solvers were greater than 100 seconds, totalling to 9 instances. The results clearly showcase that the ILP model in [11] under CPLEX fails to solve any of these hard instances optimally within the time limit, while the SAT-based approach in [13] under the solver Cadical only manages to find the optimal solution for 3 instances. In contrast, the performances of our model with binary representation under the three chosen MaxSAT solvers showcase a better efficiency, with MaxHS outperforming the others in most cases and EvalMaxSAT managing to solve all these hard instances to optimality within the allocated time.

Instance	n	m	c	OPT.	ILP	SAT	MaxSAT (inaugural)			MaxSAT (binary)		
					CPLEX	Cadical	WMaxCDCL	EvalMaxSAT	MaxHS	WMaxCDCL	EvalMaxSAT	MaxHS
ROSZIEG-5	25	6	25	161	-	473.82	842.38	1264.23	673.50	487.02	887.25	**312.75**
HESKIA-2	28	5	205	141	-	-	-	-	-	-	**2566.20**	3207.14
HESKIA-1	28	8	138	204	-	-	1334.18	894.55	598.67	662.95	202.30	**164.47**
BUXEY-5	29	8	41	245	-	1029.22	1061.14	-	3379.66	571.76	760.50	**167.43**
BUXEY-3	29	11	33	305	-	-	3223.44	-	-	-	**2394.87**	-
SAWYER30-6	30	8	41	223	-	-	-	-	-	-	1269.52	**617.74**
SAWYER30-3	30	12	30	303	-	-	2751.95	2406.47	2267.69	1259.66	940.77	**519.14**
GUNTHER-5	35	9	61	222	-	-	3022.39	3287.49	-	1571.73	1302.63	**1015.98**
WARNECKE-6	58	25	65	725	-	765.04	1980.35	2768.06	2818.22	1012.76	1870.77	**709.86**

TABLE II: Results on some hard benchmarks. Each method's running times (in seconds) are reported. For each instance, the number of tasks (n), the number of stations (m), the cycle time (c), and the obtained optimal value ($OPT.$) are listed. If an optimal value cannot be computed within 3600s, "-" is reported. The best running times are marked in bold.



(a) WMaxCDCL (b) EvalMaxSAT (c) MaxHS

Fig. 6: Runtime comparison (in seconds) of the binary and inaugural models on the logarithmic scale. When an instance cannot be solved, its corresponding solving time is regarded as the cutoff time (3600s).

C. Effect Of The Binary Representation

Finally, to conduct a more in-depth comparison between the inaugural and binary models, we present the detailed runtime comparisons for each MaxSAT solver in Figure 6. The experimental results clearly demonstrate the effectiveness of the binary representation approach proposed in Section III-B. In each subgraph, the horizontal axis represents the runtime of the binary model on the logarithmic scale while the vertical axis indicates the runtime of the inaugural model, also on the logarithmic scale. The diagonal line serves as a reference, where points above the line signify instances where the binary model achieves shorter runtimes than the inaugural model and vice-versa. Across all three subgraphs, we clearly observe that the majority of the points lie above or at the diagonal line, with the percentages of 96.91%, 78.35%, 100%, respectively for the MaxSAT solvers WMaxCDCL, EvalMaxSAT and MaxHS, indicating that overall the binary representation approach leads to faster runtimes compared to the inaugurated model. This highlights the robustness of the binary representation approach across different solvers and also underscores its potential for solving similar constraint optimization problems in other domains.

V. Conclusion

This paper proposes MaxSAT models tailored for the SALB3PM problem. We first introduce the inaugural MaxSAT model for SALBP3PM. We show that our formulation is able to outperform the ILP model in [11] and that it is competitive with the SAT-based approach introduced in [13]. Furthermore, we introduce an enhanced model based on a binary representation, enabling to reduce the search space. Our experimental results clearly showcase the efficiency and robustness of our enhanced model as it outperforms other methods in the literature. To the best of our knowledge, this is the first attempt to address the SALB3PM problem using MaxSAT. As future work, it will be interesting to expand our experimentation to a larger dataset, to further improve our model, as well as to explore the potential benefits of parameter tuning for further performance gains. In particular, it would be relevant to compute tighter bounds by taking advantage of the power of SAT solvers in effectively solving decision problems. Furthermore, our approach also has the potential to be applied in the context of other variants of SALBP such as SALBP1 and SALBP2, and the notion of binary representation might also find its application in other approaches.

REFERENCES

[1] A. Scholl, *Balancing and sequencing of assembly lines*, 2nd ed., ser. Contributions to Management Science. Heidelberg, Germany: Physica-Verlag, Feb. 1999. [Online]. Available: https://link.springer.com/book/9783790811803

[2] A. Scholl and C. Becker, "State-of-the-art exact and heuristic solution procedures for simple assembly line balancing," *Eur. J. Oper. Res.*, vol. 168, no. 3, pp. 666–693, 2006.

[3] A. Mumani, O. Abu-Farsakh, S. Obaidat, and A. Momani, "A gra-based approach for optimal selection of salbp-1 assembly line balancing heuristics," *Journal of Engineering Research*, 2023.

[4] A. Mumani, O. Abu-Farsakh, A. Momani, and S. Obaidat, "Studying the effects of operational factors on the performance of line balancing heuristics for solving salbp-1," *Arabian Journal for Science and Engineering*, vol. 48, no. 11, p. 15609–15624, May 2023. [Online]. Available: http://dx.doi.org/10.1007/s13369-023-07937-z

[5] N. Kamarudin and M. A. Rashid, "Modelling of simple assembly line balancing problem type 1 (salbp-1) with machine and worker constraints," *Journal of Physics: Conference Series*, vol. 1049, no. 1, p. 012037, jul 2018.

[6] O. Kilincci, "A petri net-based heuristic for simple assembly line balancing problem of type 2," *The International Journal of Advanced Manufacturing Technology*, vol. 46, no. 1–4, p. 329–338, May 2009. [Online]. Available: http://dx.doi.org/10.1007/s00170-009-2082-z

[7] R. Klein and A. Scholl, "Maximizing the production rate in simple assembly line balancing — a branch and bound procedure," *European Journal of Operational Research*, vol. 91, no. 2, pp. 367–385, 1996. [Online]. Available: https://www.sciencedirect.com/science/article/pii/037722179500047X

[8] R. Pitakaso, K. Sethanan, G. Jirasirilerd, and P. Golinska-Dawson, "A novel variable neighborhood strategy adaptive search for SALBP-2 problem with a limit on the number of machine's types," *Ann. Oper. Res.*, vol. 324, no. 1, pp. 1501–1525, 2023.

[9] N. Wei and I. Chao, "A solution procedure for type E simple assembly line balancing problem," *Comput. Ind. Eng.*, vol. 61, no. 3, pp. 824–830, 2011.

[10] P. T. Zacharia and A. C. Nearchou, "A meta-heuristic algorithm for the fuzzy assembly line balancing type-e problem," *Comput. Oper. Res.*, vol. 40, no. 12, pp. 3033–3044, 2013.

[11] P. Gianessi, X. Delorme, and O. Masmoudi, "Simple assembly line balancing problem with power peak minimization," in *Advances in Production Management Systems. Production Management for the Factory of the Future - IFIP WG 5.7 International Conference, APMS 2019, Austin, TX, USA, September 1-5, 2019, Proceedings, Part I*, ser. IFIP Advances in Information and Communication Technology, F. Ameri, K. E. Stecke, G. von Cieminski, and D. Kiritsis, Eds., vol. 566. Springer, 2019, pp. 239–247.

[12] D. Lamy, X. Delorme, and P. Gianessi, "Line balancing and sequencing for peak power minimization," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 10 411–10 416, 2020, 21st IFAC World Congress. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405896320335448

[13] M. Py, A. Tuyaba, L. Deroussi, N. Grangeon, and S. Norre, "Application of SAT to the Simple Assembly Line Balancing Problem with Power Peak Minimization," 15th Pragmatics of SAT international workshop, 2024.

[14] C. M. Li and F. Manyà, "MaxSAT, Hard and Soft Constraints," in *Handbook of Satisfiability - Second Edition*, ser. Frontiers in Artificial Intelligence and Applications, A. Biere, M. Heule, H. van Maaren, and T. Walsh, Eds. IOS Press, 2021, vol. 336, pp. 903–927.

[15] F. Bacchus, M. Järvisalo, and R. Martins, "Maximum satisfiabiliy," in *Handbook of satisfiability*. IOS Press, 2021, pp. 929–991.

[16] R. J. Asín Achá and R. Nieuwenhuis, "Curriculum-based course timetabling with SAT and MaxSAT," *Ann. Oper. Res.*, vol. 218, no. 1, pp. 71–91, 2014.

[17] M. Bofill, M. Garcia, J. Suy, and M. Villaret, "MaxSAT-Based Scheduling of B2B Meetings," in *Integration of AI and OR Techniques in Constraint Programming - 12th International Conference, CPAIOR 2015, Barcelona, Spain, May 18-22, 2015, Proceedings*, ser. LNCS, L. Michel, Ed., vol. 9075. Springer, 2015, pp. 65–73.

[18] E. Demirovic and N. Musliu, "MaxSAT-based large neighborhood search for high school timetabling," *Comput. Oper. Res.*, vol. 78, pp. 172–180, 2017.

[19] E. Demirovic, N. Musliu, and F. Winter, "Modeling and solving staff scheduling with partial weighted maxSAT," *Ann. Oper. Res.*, vol. 275, no. 1, pp. 79–99, 2019.

[20] X. Liao, H. Zhang, M. Koshimura, R. Huang, and W. Yu, "Maximum Satisfiability Formulation for Optimal Scheduling in Overloaded Real-Time Systems," in *PRICAI 2019: Trends in Artificial Intelligence - 16th Pacific Rim International Conference on Artificial Intelligence, Cuvu, Yanuca Island, Fiji, August 26-30, 2019, Proceedings, Part I*, ser. LNCS, A. C. Nayak and A. Sharma, Eds., vol. 11670. Springer, 2019, pp. 618–631.

[21] S. Cherif, H. Sattoutah, C. Li, C. Lucet, and L. B. Devendeville, "Minimizing working-group conflicts in conference session scheduling through maximum satisfiability (short paper)," in *30th International Conference on Principles and Practice of Constraint Programming, CP 2024, September 2-6, 2024, Girona, Spain*, ser. LIPIcs, P. Shaw, Ed., vol. 307. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024, pp. 34:1–34:11. [Online]. Available: https://doi.org/10.4230/LIPIcs.CP.2024.34

[22] R. Asín, R. Nieuwenhuis, A. Oliveras, and E. Rodríguez-Carbonell, "Cardinality Networks and Their Applications," in *Theory and Applications of Satisfiability Testing - SAT 2009, 12th International Conference, SAT 2009, Swansea, UK, June 30 - July 3, 2009. Proceedings*, ser. LNCS, O. Kullmann, Ed., vol. 5584. Springer, 2009, pp. 167–180.

[23] O. Roussel and V. Manquinho, "Pseudo-boolean and cardinality constraints," in *Handbook of satisfiability*. IOS Press, 2021, pp. 1087–1129.

[24] F. B. Talbot, J. H. Patterson, and W. V. Gehrlein, "A comparative evaluation of heuristic line balancing techniques," *Management science*, vol. 32, no. 4, pp. 430–454, 1986.

[25] T. R. Hoffmann, "Assembly line balancing: a set of challenging problems," *The International Journal of Production Research*, vol. 28, no. 10, pp. 1807–1815, 1990.

[26] A. Scholl, *Balancing and sequencing of assembly lines*, 2nd ed. Heidelberg: Physica, 1999.

[27] IBM Corporation, *IBM ILOG CPLEX Optimization Studio*, 2022. [Online]. Available: https://www.ibm.com/analytics/cplex-optimizer

[28] A. Biere, K. Fazekas, M. Fleury, and M. Heisinger, "CaDiCaL, Kissat, Paracooba, Plingeling and Treengeling entering the SAT Competition 2020," in *Proc. of SAT Competition 2020 – Solver and Benchmark Descriptions*, ser. Department of Computer Science Report Series B, T. Balyo, N. Froleyks, M. Heule, M. Iser, M. Järvisalo, and M. Suda, Eds., vol. B-2020-1. University of Helsinki, 2020, pp. 51–53.

[29] J. Davies and F. Bacchus, "Solving MAXSAT by Solving a Sequence of Simpler SAT Instances," in *Principles and Practice of Constraint Programming - CP 2011 - 17th International Conference, CP 2011, Perugia, Italy, September 12-16, 2011. Proceedings*, ser. LNCS, J. H. Lee, Ed., vol. 6876. Springer, 2011, pp. 225–239.

[30] F. Avellaneda, "Evalmaxsat 2023," *MaxSAT Evaluation 2022: Solver and Benchmark Descriptions*, pp. 12–13, 2023. [Online]. Available: https://helda.helsinki.fi/server/api/core/bitstreams/3e45b423-573c-447f-8770-4e7785bfa38b/content#page=13

[31] C. Li, Z. Xu, J. Coll, F. Manyà, D. Habet, and K. He, "Boosting branch-and-bound MaxSAT solvers with clause learning," *AI Commun.*, vol. 35, no. 2, pp. 131–151, 2022.

[32] C. M. Li, Z. Xu, J. Coll, F. Manyà, D. Habet, and K. He, "Combining clause learning and branch and bound for maxsat," in *27th International Conference on Principles and Practice of Constraint Programming, CP 2021, Montpellier, France (Virtual Conference), October 25-29, 2021*, ser. LIPIcs, L. D. Michel, Ed., vol. 210. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021, pp. 38:1–38:18. [Online]. Available: https://doi.org/10.4230/LIPIcs.CP.2021.38

[33] A. Ignatiev, A. Morgado, and J. Marques-Silva, "PySAT: A Python Toolkit for Prototyping with SAT Oracles," in *Theory and Applications of Satisfiability Testing - SAT 2018 - 21st International Conference, Held as Part of the Federated Logic Conference, FloC 2018, Oxford, UK, July 9-12, 2018, Proceedings*, ser. LNCS, O. Beyersdorff and C. M. Wintersteiger, Eds., vol. 10929. Springer, 2018, pp. 428–437.

[34] N. Manthey, T. Philipp, and P. Steinke, "A more compact translation of pseudo-boolean constraints into CNF such that generalized arc consistency is maintained," in *KI 2014: Advances in Artificial Intelligence - 37th Annual German Conference on AI, Stuttgart, Germany, September 22-26, 2014. Proceedings*, ser. LNCS, C. Lutz and M. Thielscher, Eds., vol. 8736. Springer, 2014, pp. 123–134.