



**HAL**  
open science

# Utilisation de SAT pour résoudre le problème SALBP avec minimisation du pic de consommation

Matthieu Py, Arnauld Tuyaba

► **To cite this version:**

Matthieu Py, Arnauld Tuyaba. Utilisation de SAT pour résoudre le problème SALBP avec minimisation du pic de consommation. Journées Francophones de Programmation par Contraintes (JFPC), Jul 2023, Strasbourg, France. hal-04318091

**HAL Id: hal-04318091**

**<https://hal.science/hal-04318091v1>**

Submitted on 1 Dec 2023

**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.

# Utilisation de SAT pour résoudre le problème SALBP avec minimisation du pic de consommation

Matthieu Py, Arnaud Tuyaba

Université Clermont Auvergne, Mines Saint-Etienne, CNRS, LIMOS, F-63000 Clermont-Ferrand, France

{matthieu.py, arnaud.tuyaba}@uca.fr

## Résumé

Dans cet article, on s'intéresse au problème SALB3PM (Simple Assembly Line Balancing Problem with Power Peak Minimization). On propose une résolution de ce problème en utilisant une séquence de problèmes SAT à résoudre, avec appels à un solveur SAT pour résoudre chaque sous-problème. La pertinence de cette approche est vérifiée expérimentalement par comparaison avec les résultats existants sur quelques instances issues de la littérature. Ces travaux sont une extension de [4].

## Mots-clés

Ordonnancement, SALB3PM, Modélisation, Programmation par contraintes, SAT, contraintes énergétiques

## 1 Présentation du problème SALB3PM

Le problème SALB3PM (Simple Assembly Line Balancing Problem with Power Peak Minimization) est un problème d'optimisation qui consiste, étant donné un ensemble de tâches et un ensemble de machines, à construire l'ordonnancement des tâches sur les différentes machines de manière à minimiser le pic d'énergie consommé par l'ordonnancement obtenu. Les données du problème SALB3PM sont les suivantes :

- Un ensemble  $O$  de  $n$  tâches, où chaque tâche  $j \in O$  a une durée de traitement  $t_j$  et une consommation énergétique  $W_j$ .
- Un ensemble de machines  $M$  numérotées de 1 à  $m$ .
- Un temps de cycle  $c$  : après  $c$  unités de temps, les machines sont arrêtées. L'horizon temporel est découpé en périodes et est noté  $T = \{0, \dots, c-1\}$ . On note l'ensemble  $T^j = \{0, \dots, c-t_j\}$  l'ensemble des périodes où peut commencer la tâche  $j \in O$  pour pouvoir terminer avant la fin du temps de cycle.
- Un ensemble de précédences  $P$  : si une tâche  $i \in O$  précède une tâche  $j \in O$ , ce qui est noté  $i \prec j$ , alors :
  - soit la tâche  $i$  est affectée à une machine de plus petit numéro que la tâche  $j$ ,
  - soit les tâches  $i$  et  $j$  sont affectées à la même machine mais la tâche  $i$  est ordonnancée avant la tâche  $j$ .

L'objectif du problème SALB3PM est de décider d'affecter chaque tâche sur une machine et d'ordonnancer les tâches affectées sur les mêmes machines (c'est à dire déterminer leur période de lancement), en minimisant le pic d'énergie de la solution obtenue, c'est à dire en minimisant la consommation maximale énergétique des tâches ordonnancées simultanément.

## 2 Modélisation linéaire existante

Dans l'article [3], le problème SALB3PM est modélisé sous la forme du programme linéaire en nombres entiers présenté dans la figure 1. Les variables de décision sont les suivantes :

- Les variables d'affectation  $X_{j,k}$ . Étant donnée une tâche  $j \in O$  et une machine  $k \in M$ , la variable  $X_{j,k}$  vaut 1 si et seulement si la tâche  $j$  est affectée à la machine  $k$ .
- Les variables d'ordonnancement  $S_{j,t}$ . Étant donnée une tâche  $j \in O$  et une période  $t \in T$ , la variable  $S_{j,t}$  vaut 1 si et seulement si la tâche  $j$  débute à la période  $t$ .
- La variable  $W_{max}$  est une borne supérieure sur le pic de consommation énergétique.

Les lignes du programme linéaire en nombres entiers modélisent la fonction objectif et les contraintes du problème :

1. On doit minimiser le pic de consommation énergétique.
2. Chaque tâche doit être affectée à exactement une machine.
3. La somme des durées des tâches affectées à chaque machine ne peut pas dépasser le temps de cycle.
4. Si une tâche  $i$  précède une tâche  $j$  ( $i \prec j$ ), alors la tâche  $i$  ne peut pas être affectée à une machine de plus grand numéro que la tâche  $j$ .
5. Chaque tâche doit débiter à une et une seule période.
6. Si une tâche  $i$  précède une tâche  $j$  ( $i \prec j$ ) et que les deux tâches sont affectées à la même machine, alors la tâche  $j$  débute forcément après la tâche  $i$ .
7. Il est impossible d'avoir deux tâches en cours d'exécution sur la même machine à une période donnée.

$$\begin{aligned}
\min \quad & W_{max} & (1) \\
s.t. \quad & \sum_{k \in M} X_{j,k} = 1 & \forall j \in O \quad (2) \\
& \sum_{j \in O} t_j \times X_{j,k} \leq c & \forall k \in M \quad (3) \\
& X_{j,k} \leq \sum_{h \in M: h \leq k} X_{i,h} & \forall i, j \in O : i \prec j, k \in M \quad (4) \\
& \sum_{t \in T^j} S_{j,t} = 1 & \forall j \in O \quad (5) \\
& S_{j,t} \leq \sum_{\tau=0}^{t-t_i} S_{i,\tau} + 2 - X_{i,k} - X_{j,k} & \forall i, j \in O : i \prec j, k \in M, t \in T^j \quad (6) \\
& X_{i,k} + X_{j,k} + \sum_{\tau=t-t_i+1}^t S_{i,\tau} + \sum_{\tau=t-t_j+1}^t S_{j,\tau} \leq 3 & \forall i, j \in O, k \in M, t \in T \quad (7) \\
& \sum_{j \in O} W_j \times \left( \sum_{\tau=t-t_j+1}^t S_{j,\tau} \right) \leq W_{max} & \forall t \in T \quad (8) \\
& X_{i,k}, S_{i,t} \in \{0, 1\}, W_{max} \in \mathbb{Z}^+ & \forall i \in O, k \in M, t \in T
\end{aligned}$$

FIGURE 1 – Modélisation de SALB3PM sous forme de PLNE [3]

8. Le pic de consommation énergétique est plus grand que la consommation maximale énergétique des tâches ordonnancées sur une même période.

### 3 Approche SAT

On modélise le problème SALB3PM sous forme d'un problème de satisfaisabilité booléenne (problème SAT). Comme le problème SAT est un problème de décision, et non un problème d'optimisation, on modélise initialement le problème de faisabilité qui consiste à trouver une solution qui respecte toutes les contraintes du problème et qui est présenté dans la figure 2. Les variables de décisions sont les suivantes, où les deux premiers jeux de variables de décision sont les mêmes que pour la modélisation linéaire :

- Les variables d'affectation  $X_{j,k}$ . Étant donnée une tâche  $j \in O$  et une machine  $k \in M$ , la variable  $X_{j,k}$  vaut 1 si et seulement si la tâche  $j$  est affectée à la machine  $k$ .
- Les variables d'ordonnancement  $S_{j,t}$ . Étant donnée une tâche  $j \in O$  et une période  $t \in T$ , la variable  $S_{j,t}$  vaut 1 si et seulement si la tâche  $j$  débute à la période  $t$ .
- Les variables d'activité  $A_{j,t}$ . Étant donnée une tâche  $j \in O$  et une période  $t \in T$ , la variable  $A_{j,t}$  vaut 1 si et seulement si la tâche  $j$  est active à la période  $t$ .

Les premières lignes de la formule propositionnelle modélisent les contraintes du problème :

1. Chaque tâche doit être affectée à au moins une machine.
2. Chaque tâche doit être affectée à au plus une machine.

3. Si une tâche  $i$  précède une tâche  $j$  ( $i \prec j$ ), alors la tâche  $i$  ne peut pas être affectée à une machine de plus grand numéro que la tâche  $j$ .
4. Chaque tâche doit débiter à au moins une période.
5. Chaque tâche doit débiter à au plus une période.
6. Une tâche ne peut pas débiter si elle ne peut pas terminer avant la fin du temps de cycle.
7. Il est impossible d'avoir deux tâches en cours d'exécution sur la même machine à une période donnée.
8. Si une tâche débute à une période donnée, alors elle est forcément active ensuite pendant la durée de son exécution.
9. Si une tâche  $i$  précède une tâche  $j$  ( $i \prec j$ ), alors la tâche  $j$  ne peut pas être affectée à la même machine  $i$  et commencer après le début de la tâche  $i$ .

On peut constater, par exemple en comparant la ligne (2) du programme linéaire et les lignes (2) et (3) de la formule propositionnelle, que le modèle SAT est beaucoup plus gros que le modèle linéaire. Pour cette raison, on effectue un prétraitement pour calculer un ensemble de décisions *évidentes* à cause de la liste des précédences et de la durée des tâches. Ces prétraitements servent à rajouter de nouvelles contraintes permettent de réduire la taille de l'espace de recherche mais elles servent surtout à éliminer des contraintes du modèle initial (si une des contraintes (2) à (9) est en contradiction avec une des contraintes (10) à (12), alors elle n'est pas intégrée au modèle) :

- 10 Si une chaîne de précédences empêche une tâche  $j$  d'être affectée sur une machine  $k$ , on interdit cette affectation (on le note  $ip(j, k) = 1$ )

$$\begin{aligned}
& \bigvee_{k \in M} X_{j,k} && \forall j \in O \quad (1) \\
& \overline{X_{j,k_1}} \vee \overline{X_{j,k_2}} && \forall j \in O, k_1, k_2 \in M : k_1 \neq k_2 \quad (2) \\
& \overline{X_{j,k}} \vee \overline{X_{i,h}} && \forall i, j \in O : i \prec j, k, h \in M : k < h \quad (3) \\
& \bigvee_{t \in T^j} S_{j,t} && \forall j \in O \quad (4) \\
& \overline{S_{j,t_1}} \vee \overline{S_{j,t_2}} && \forall j \in O, t_1, t_2 \in T^j : t_1 \neq t_2 \quad (5) \\
& \overline{S_{j,t}} && \forall j \in O, t \in T : t \notin T^j \quad (6) \\
& \overline{X_{i,k}} \vee \overline{X_{j,k}} \vee \overline{A_{i,t}} \vee \overline{A_{j,t}} && \forall i, j \in O : i \neq j, k \in M, t \in T \quad (7) \\
& \overline{S_{j,t}} \vee \overline{A_{j,t+\epsilon}} && \forall j \in O, t \in T^j, \epsilon \in [0, t_j - 1] \quad (8) \\
& \overline{X_{i,k}} \vee \overline{X_{j,k}} \vee \overline{S_{i,t_1}} \vee \overline{S_{j,t_2}} && \forall i, j \in O : i \prec j, k \in M, \\
& && t_1 \in T^i, t_2 \in T^j : t_1 > t_2 \\
& \overline{X_{j,k}} && \forall j \in O, k \in M : ip(j, k) \quad (10) \\
& \overline{X_{j,k}} \vee \overline{S_{j,t}} && \forall j \in O, k \in M : p(j, k), t \in T^j : ip(j, k, t) \quad (11) \\
& A_{j,t} && \forall t \in [c - t_i, t_i - 1] \quad (12) \\
& \bigvee_{j \in C} \overline{A_{j,t}} && \forall C \in \mathbb{C}, t \in T \quad (13)
\end{aligned}$$

FIGURE 2 – Modélisation des contraintes de SALB3PM sous forme de formule propositionnelle

- 11 Si une chaîne de précédences empêche une tâche  $j$  d'être affectée sur une machine  $k$  à une période  $t$ , on interdit d'avoir simultanément cette affectation et cet ordonnancement (on le note  $ip(j, k, t) = 1$ )
- 12 Si une tâche a une durée qui dépasse la moitié du temps de cycle, elle est forcément active aux dates médianes

Une fois que l'on a écrit le modèle mathématique précédent, sa résolution par un algorithme de résolution du problème SAT permet de trouver un ordonnancement faisable.

Pour ensuite déterminer l'ordonnancement avec un pic d'énergie minimum, on relance, autant de fois que possible, notre algorithme en intégrant au modèle des contraintes supplémentaires. Ces contraintes correspondent à l'interdiction d'exécuter en même temps un ensemble de tâches qui l'étaient dans la dernière solution calculée et dont la somme des énergies est supérieure au pic d'énergie de la meilleure solution rencontrée depuis le début de l'exécution de l'algorithme. Ces contraintes correspondent à la ligne (13) du modèle et l'ensemble  $\mathbb{C}$  contient en permanence l'ensemble des ensembles de tâches que l'on ne souhaite plus voir ordonnancées en même temps. Après ajout de ces contraintes, on relance la résolution du problème de faisabilité.

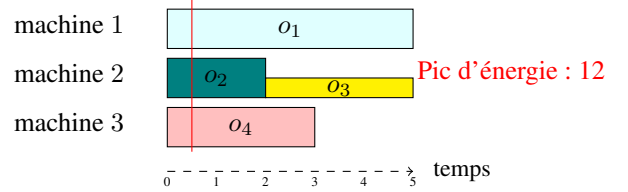
Lorsqu'il n'est plus possible de trouver une solution pour le problème de faisabilité, la meilleure solution rencontrée jusqu'à cet instant est une solution optimale pour le problème SALB3PM.

## 4 Exemple illustratif

On considère 4 tâches, 3 machines, un temps de cycle  $c = 5$  et des précédences lexicographiques  $o_1 \prec o_2 \prec o_3 \prec o_4$  :

Tâche	$o_1$	$o_2$	$o_3$	$o_4$
Durée	5	2	3	3
Energie	4	4	2	4

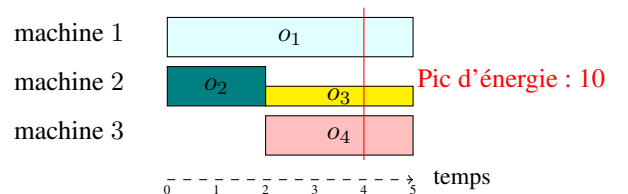
Tout d'abord, on crée le modèle initial et, à la première itération, le solveur SAT trouve la solution suivante :



La solution proposée a un pic d'énergie de 12, causé par l'ordonnancement des tâches  $o_1$ ,  $o_2$  et  $o_4$ . On ajoute donc les contraintes suivantes au modèle, pour empêcher désormais ces trois tâches d'être actives à une même période :

$$\overline{A_{o_1,t}} \vee \overline{A_{o_2,t}} \vee \overline{A_{o_4,t}} \quad \forall t \in T$$

On relance ensuite le solveur SAT et, à la deuxième itération, le solveur SAT trouve la solution suivante :



Instance	Données			Résultat PLNE			Résultat SAT		
	$n$	$m$	$c$	Pic	Temps	Statut	Pic	Temps	Statut
mertens-1	7	6	6	<b>104</b>	<b>0.01 s</b>	Optimal	<b>104</b>	0.227 s	Optimal
mertens-2	7	2	18	<b>49</b>	<b>0.09 s</b>	Optimal	<b>49</b>	0.238 s	Optimal
bowman-1	8	5	20	<b>164</b>	<b>0.05 s</b>	Optimal	<b>164</b>	0.226 s	Optimal
jaeschke-1	9	8	6	<b>248</b>	<b>0.02 s</b>	Optimal	<b>248</b>	0.21 s	Optimal
jaeschke-2	9	3	18	<b>78</b>	<b>0.16 s</b>	Optimal	<b>78</b>	0.279 s	Optimal
jackson-1	11	8	7	<b>179</b>	<b>0.08 s</b>	Optimal	<b>179</b>	0.254 s	Optimal
jackson-2	11	2	94	<b>65</b>	1.05 s	Optimal	<b>65</b>	<b>0.502 s</b>	Optimal
mansoor-1	11	4	48	<b>145</b>	0.69 s	Optimal	<b>145</b>	<b>0.379 s</b>	Optimal
mansoor-2	11	2	94	<b>77</b>	5.03 s	Optimal	<b>77</b>	<b>0.563 s</b>	Optimal
mitchell-1	21	8	14	<b>221</b>	1.44 s	Optimal	<b>221</b>	<b>0.6 s</b>	Optimal
mitchell-2	21	3	39	<b>85</b>	427.94 s	Optimal	<b>85</b>	<b>9.884 s</b>	Optimal
roszieg-1	25	10	14	<b>254</b>	104.36 s	Optimal	<b>254</b>	<b>7.8 s</b>	Optimal
roszieg-2	25	4	32	<b>117</b>	163.59 s	Optimal	<b>117</b>	<b>5.653 s</b>	Optimal
heskiaoff-1	28	8	138	$\leq 290$	$\geq 3600$ s		<b>251</b>	<b>196.693 s</b>	Optimal
heskiaoff-2	28	3	342		$\geq 3600$ s		$\leq 107$	$\geq 3600$ s	
buxey-1	29	14	25	$\leq 292$	$\geq 3600$ s		$\leq 292$	$\geq 3600$ s	
buxey-2	29	7	47	$\leq 350$	$\geq 3600$ s		<b>172</b>	<b>137.46 s</b>	Optimal
sawyer-1	30	14	25	<b>395</b>	157 s	Optimal	<b>395</b>	<b>2.044 s</b>	Optimal
sawyer-2	30	7	47		$\geq 3600$ s		<b>214</b>	<b>257.916 s</b>	Optimal
gunther-1	35	14	40	<b>394</b>	2297 s	Optimal	<b>394</b>	<b>2.131 s</b>	Optimal
gunther-2	35	9	54		$\geq 3600$ s		<b>295</b>	<b>6.081 s</b>	Optimal

TABLE 1 – Comparaison des approches PLNE et SAT sur quelques instances

La solution proposée a un pic d'énergie de 10, causé par l'ordonnancement des tâches  $o_1$ ,  $o_3$  et  $o_4$ . On ajoute donc les contraintes suivantes au modèle, pour empêcher désormais ces trois tâches d'être actives à une même période :

$$\overline{A_{o_1,t}} \vee \overline{A_{o_3,t}} \vee \overline{A_{o_4,t}} \quad \forall t \in T$$

On relance ensuite le solveur SAT et, à la troisième itération, le solveur SAT trouve que le problème ne possède pas de solution. La meilleure solution proposée jusqu'à maintenant, qui est ici la solution proposée à la deuxième itération, est donc la solution optimale, avec un pic d'énergie de 10.

## 5 Expérimentations

Nous avons expérimenté notre algorithme sur plusieurs instances de la littérature, avec pour chaque instance les résultats obtenus en implémentant le modèle existant sur CPLEX [3] et les résultats obtenus avec notre algorithme. Le problème SAT sous-jacent est résolu grâce au solveur Sat4j [1]. On donne, pour chaque instance, le nombre de tâches  $n$ , de machines  $m$ , le temps de cycle  $c$  et, pour chaque méthode, le pic d'énergie de la meilleure solution calculée, la durée d'exécution (au plus 1 heure) et son statut (optimal ou non). On constate dans la table 1 que l'approche proposée permet de trouver la solution optimale sur plus d'instances (19 instances sur 21) qu'avec l'approche ILP connue (15 sur 21).

## 6 Conclusion

Dans cet article, on a étudié le problème SALB3PM. On a proposé une approche qui fait appel itérativement au problème SAT pour résoudre ce problème. Cette approche a été comparée avec une approche par programmation linéaire en

nombre entiers existante, et donne des résultats préliminaires prometteurs. Ces résultats préliminaires doivent être confirmés en offrant le même niveau de finesse de modélisation pour les deux approches et en comparant avec les récentes améliorations du modèle linéaire [2]. Il faudra réfléchir également à comment réduire la taille du modèle SAT initial et comment mieux gérer la fonction objectif. Enfin, il serait intéressant d'essayer d'hybrider ces approches entre elles ou avec d'autres approches heuristiques [5].

## Références

- [1] Daniel Le Berre and Anne Parrain. The Sat4j library, release 2.2. *Journal on Satisfiability, Boolean Modeling and Computation*, 7 :59–64, 2010.
- [2] Paolo Gianessi and Xavier Delorme. A new ILP Model for a Line Balancing Problem with Minimization of Power Peak. In *31st European Conference on Operational Research (EURO-2021)*, 2021.
- [3] Paolo Gianessi, Xavier Delorme, and Oussama Masmoudi. Simple Assembly Line Balancing Problem with Power Peak Minimization. In *IFIP International Conference on Advances in Production Management Systems (APMS)*, 2019.
- [4] Matthieu Py and Arnauld Tuyaba. Résolution du problème SALB3PM grâce à une approche SAT. In *24ème édition du congrès annuel de la Société Française de Recherche Opérationnelle et d'Aide à la Décision (ROADEF)*, 2023.
- [5] Arnauld Tuyaba, Laurent Deroussi, Nathalie Grangeon, and Sylvie Norre. Prise en compte de la consommation énergétique dans l'équilibrage de lignes d'assemblage. In *24ème édition du congrès annuel de la Société Française de Recherche Opérationnelle et d'Aide à la Décision (ROADEF)*, 2023.