



AVERTISSEMENT

Ce document est le fruit d'un long travail approuvé par le jury de soutenance et mis à disposition de l'ensemble de la communauté universitaire élargie.

Il est soumis à la propriété intellectuelle de l'auteur. Ceci implique une obligation de citation et de référencement lors de l'utilisation de ce document.

D'autre part, toute contrefaçon, plagiat, reproduction illicite encourt une poursuite pénale.

Contact : ddoc-theses-contact@univ-lorraine.fr

LIENS

Code de la Propriété Intellectuelle. articles L 122. 4

Code de la Propriété Intellectuelle. articles L 335.2- L 335.10

http://www.cfcopies.com/V2/leg/leg_droi.php

<http://www.culture.gouv.fr/culture/infos-pratiques/droits/protection.htm>

THESE

Présentée pour l'obtention du grade de
Docteur de l'Université de Lorraine
en Automatique, traitement du signal et des images, génie informatique
par

Antoine MANCEAUX

**Contribution au rééquilibrage dynamique des lignes
d'assemblage : modélisation, résolutions et applications.**

Soutenue publiquement le 11/12/2015 devant le jury composé de :

<i>RAPPORTEURS :</i>	MR. YANNICK FREIN	PROFESSEUR À L'I.N.P. DE GRENOBLE
	MR. ARMAND TOGUYÉNI	PROFESSEUR À L'ÉCOLE CENTRALE DE LILLE
<i>EXAMINATEURS :</i>	MR. YVES SALLEZ	MAÎTRE DE CONFÉRENCES HDR À L'UNIVERSITÉ DE VALENCIENNES
	MR. JEAN-FANÇOIS PÉTIN	PROFESSEUR À L'UNIVERSITÉ DE LORRAINE
	MR ANDRÉ THOMAS	PROFESSEUR À L'UNIVERSITÉ DE LORRAINE (DIRECTEUR DE LA THÈSE)
	MME. HIND BRIL EL-HAOUZI	MAÎTRE DE CONFÉRENCES À L'UNIVERSITÉ DE LORRAINE (CO-DIRECTEUR)
<i>INVITÉ :</i>	MR. THIERRY MUNIER	RESPONSABLE INDUSTRIEL, ENTREPRISE TRANE



Remerciements

Je souhaite tout d'abord remercier Monsieur Armand Toguyéni, professeur à l'École Centrale de Lille et Monsieur Yannick Frein, professeur à l'Institut National Polytechnique de Grenoble et à l'École Nationale Supérieure de Génie Industriel pour m'avoir fait l'honneur de considérer mes travaux de thèse en tant que rapporteurs. Leur lecture attentive a permis des discussions pertinentes et enrichissantes.

Mes remerciements s'adressent également à Monsieur Yves Sallez, maître de conférences HDR à l'Université de Valenciennes et du Hainaut Cambrésis pour avoir accepté d'étudier mes travaux en temps que membre du jury.

Je tiens également à remercier mes encadrants de thèse Monsieur André Thomas, Madame Hind Bril El-Haouzi et Monsieur Jean-François Pétin pour leurs précieux conseils, leur soutien et leur patience. Leurs compétences complémentaires ont été une des clés dans l'aboutissement de cette thèse et je leur en suis reconnaissant.

Je remercie également Monsieur Thierry Munier et Stéphane Hacquard ainsi que les collègues de l'entreprise TRANE pour leurs encouragements et pour leur aide lors des différents projets réalisés au sein de l'entreprise. J'ai pu alors en apprendre autant sur l'entreprise que sur moi-même.

Mes remerciements s'adressent également à l'ensemble des membres du CRAN, pour les fructueuses discussions et leur chaleureux accueil au sein du laboratoire.

Je souhaite remercier mes amis pour leur soutien de tous les instants que ce soit à Épinal ou à distance.

Enfin, ma gratitude va à mes parents et à mes proches pour m'avoir soutenu pendant toutes ces années.

Table des Matières

Liste des Figures.....	9
Index des tableaux.....	10
Introduction générale.....	11
Chapitre 1 : Contextes industriel et scientifique.....	15
I. Introduction.....	17
II. Objet d'étude : système de production & DFT.....	17
II.1. La DFT.....	17
II.2. Le takt time, la « voix du client » et outil de pilotage.....	18
II.2.1. Conception des lignes d'assemblage.....	19
II.2.2. Équilibrage des lignes d'assemblage.....	19
III. Problématique industrielle : le réactif en support du prédictif.....	21
III.1. Les variations à moyen ou/et long termes :.....	21
III.1.1. Description.....	21
III.1.2. Leviers d'action.....	22
III.2. Les variations à court terme:.....	23
III.2.1. Description.....	23
III.2.2. Leviers d'action.....	24
III.3. Proposition d'un équilibrage prédictif et réactif.....	25
IV. Contexte de recherche.....	26
IV.1. Les différents type de problèmes ALB.....	26
IV.1.1. Introduction.....	26
IV.1.2. Définitions des problèmes d'équilibrage des lignes d'assemblage.....	27
IV.1.3. Classifications des SALBP.....	28
IV.1.4. Classifications des GALBP.....	28
IV.1.5. La reconfiguration ALBP.....	29
IV.2. Problème scientifique.....	30
IV.3. Formalisation.....	31
IV.3.1. Éléments du problème.....	31
IV.3.2. La prise en compte de l'aléa.....	31
IV.3.3. Objectif.....	32
IV.3.4. Contraintes d'allocation.....	32

IV.3.5. Contraintes de précédence.....	32
IV.3.6. Contraintes de parallélisation.....	33
V. Conclusion.....	33
Chapitre 2 : Les méthodes de résolution des ALBP.....	35
I. Introduction.....	37
II. État de l'art sur les méthodes de résolution ALBP.....	38
II.1. Méthodes exactes.....	38
II.1.1. Programmation dynamique.....	38
II.1.2. Procédure par séparation et évaluation.....	38
II.2. Méthodes approchées.....	39
II.2.1. Méthodes constructives.....	40
II.2.2. Méthodes incrémentales.....	40
II.3. Rééquilibrage des lignes d'assemblage.....	41
II.4. Conclusion de l'état de l'art.....	43
III. « Iterated Local Search » (ILS).....	43
III.1. Présentation du principe.....	43
III.2. Les différents types de voisinage et les critères d'arrêt.....	44
III.2.1. Types de voisinage.....	44
III.2.2. Critères d'arrêt.....	44
III.3. Formalisation du problème.....	45
III.3.1. Variables et paramètres du modèle.....	45
III.3.2. Contraintes.....	45
III.3.3. Objectif.....	45
III.3.4. Les différents types de voisinages.....	46
III.3.5. Présentation des perturbations.....	46
III.4. L'algorithme.....	46
IV. Recherche d'atteignabilité sur un réseau d'automates.....	48
IV.1. Introduction.....	48
IV.2. Historique des méthodes utilisant la théorie des SED.....	48
IV.3. Présentation du formalisme utilisé.....	49
IV.3.1. Les automates.....	49
IV.3.2. Un réseau d'automates.....	50
IV.4. Modélisation du problème de rééquilibrage.....	52
IV.4.1. Les tâches.....	52
IV.4.2. Les postes de travail.....	52
IV.4.3. La ligne d'assemblage.....	53

IV.5. La recherche d'atteignabilité.....	53
IV.5.1. Principe.....	53
IV.5.2. Définition de la propriété.....	55
V. Comparaison des 2 méthodes.....	55
V.1. Présentation du contexte et des instances.....	55
V.2. Évaluation des instances.....	58
V.2.1. Les indicateurs.....	58
V.2.2. Valeur des indicateurs.....	58
V.3. Évaluation des résultats obtenus.....	61
V.3.1. Les éléments d'évaluation.....	61
V.3.2. Scénario 1 :.....	62
V.3.3. Scénario 2 :.....	63
V.3.4. Scénario 3 :.....	64
V.3.5. Scénario 4 :.....	66
V.4. Analyse des résultats.....	66
VI. Conclusion.....	67
Chapitre 3 : Proposition d'une méthodologie modulaire pour la résolution des ALBP.....	69
I. Introduction.....	71
II. Présentation générale de la méthodologie.....	71
III. Classe de problème : Définition d'une bibliothèque d'opérateurs élémentaires génériques.....	73
III.1. Présentation.....	73
III.2. Analyse fonctionnelle.....	74
III.3. Conception des modules réalisant les leviers d'action.....	76
III.3.1. Conception de l'architecture des modules.....	76
III.3.2. Conception technique des modules.....	77
III.4. Implémentation des modules avec UPPAAL.....	80
III.4.1. Choix du logiciel.....	80
III.4.2. Implémentation des modules.....	81
IV. Problème spécifique : Instanciation des opérateurs génériques.....	81
IV.1. Choix des leviers et de la stratégie de reconfiguration.....	81
IV.2. Conception : assemblage des modules techniques.....	82
IV.3. Implémentation des modules avec le logiciel UPPAAL.....	83
IV.3.1. Implémentation.....	83
V. Conclusion.....	85
Chapitre 4 : Application de la proposition.....	87

I. Introduction.....	89
II. Présentation générale des cas d'étude.....	89
<i>II.1. L'entreprise.....</i>	<i>89</i>
<i>II.2. Les cas d'étude.....</i>	<i>90</i>
III. Premier cas d'étude.....	92
<i>III.1. Description.....</i>	<i>92</i>
<i>III.2. Application de la méthode.....</i>	<i>92</i>
IV. Deuxième cas d'étude.....	94
<i>IV.1. Description.....</i>	<i>94</i>
<i>IV.2. Application de la méthode.....</i>	<i>94</i>
<i>IV.2.1. Phase d'analyse.....</i>	<i>94</i>
<i>IV.2.2. Phase de conception.....</i>	<i>95</i>
<i>IV.2.3. Phase d'implémentation.....</i>	<i>96</i>
<i>IV.3. Résolution.....</i>	<i>96</i>
V. Validation de la méthode proposée par une analyse du troisième cas d'étude.....	97
<i>V.1. Description du cas d'étude.....</i>	<i>97</i>
<i>V.2. Description des scénarios et des facteurs étudiés.....</i>	<i>98</i>
<i>V.3. L'analyse.....</i>	<i>99</i>
<i>V.3.1. Analyse des effets des facteurs sur le temps de résolution.....</i>	<i>100</i>
<i>V.3.2. Analyse des effets des facteurs sur le nombre de changements.....</i>	<i>102</i>
VI. Conclusion.....	103
Conclusion générale et perspectives.....	105
Bibliographie.....	107
Annexe 1 : Implémentation de la méthode ILS.....	115
Annexe 2 : Implémentation de la méthode proposée.....	126
Annexe 3 : Applications développées pour l'entreprise Trane.....	133
Annexe 4 : Données du troisième cas d'étude.....	135
Annexe 5 : Résultats du troisième cas d'étude.....	137

Liste des Figures

Figure 1: Ligne de production et ses feeders.....	19
Figure 2: Diagramme Yamazumi représentant un équilibrage initial.....	21
Figure 3: Exemple d'un "Takt playbook" simplifié.....	23
Figure 4: Interaction entre les calculs prédictifs et les aléas.....	26
Figure 5: Exemple d'un graphe de précédence.....	27
Figure 6: Les différents types de reconfiguration de ligne et leurs horizons temporels.....	42
Figure 7: Principe de fonctionnement de ILS.....	44
Figure 8: Algorithme de la méthode ILS.....	47
Figure 9: Exemple de l'évolution synchrone de deux automates communicants.....	51
Figure 10: Modèle générique d'une tâche.....	52
Figure 11: Modèle générique d'un poste de travail.....	53
Figure 12: Modèle d'une ligne d'assemblage de 3 postes de travail et 6 tâches.....	54
Figure 13: L'organisation du feeder échangeur.....	56
Figure 14: Équilibrage initial pour chacune des 3 instances.....	57
Figure 15: Répartition des durées des tâches de la ligne 1.....	59
Figure 16: Répartition des durées des tâches de (Rashboogari and Kermanshah, 2012).....	60
Figure 17: Répartition des durées des tâches de (Dolgui, 2005).....	60
Figure 18 : Les étapes principales de la méthode proposée.....	71
Figure 19: Les étapes de notre proposition méthodologique.....	72
Figure 20: Composition de modules formant les leviers d'action.....	76
Figure 21: Interaction entre les différents types de modules.....	77
Figure 22: Modèle générique d'un module maître.....	78
Figure 23: Modèle générique d'un module d'action.....	79
Figure 24: Modèle générique d'un module de vérification.....	79
Figure 25: Heuristique de la stratégie de rééquilibrage.....	82
Figure 26: Levier « déplacement » : 1 module de sélection, 1 module d'action et 1 module de vérification. 83	
Figure 27: Levier « parallélisation » : 1 module de sélection, 2 modules d'action et 2 modules de vérification.	84
Figure 28: Résolution d'une instance par un script python.....	85
Figure 29: Aperçu des produits et des services proposés par l'entreprise Trane.....	89
Figure 30: Un « fancoil » Trane.....	90
Figure 31: Les différences entre les cas d'étude étudiés.....	91
Figure 32: Équilibrage initial du premier cas d'étude.....	92
Figure 33: Modèle du levier de déplacement.....	93
Figure 34: Équilibrage initial du deuxième cas d'étude.....	94
Figure 35: Stratégie de reconfiguration du deuxième cas d'étude.....	95
Figure 36: Exemple de résolution du deuxième cas.....	96
Figure 37: Équilibrage initial du troisième cas d'étude pour l'analyse.....	97
Figure 38: Représentation graphique des effets principaux sur la durée de résolution.....	100
Figure 39: Diagramme des interactions pour sur la durée de résolution.....	100
Figure 40: Analyse statistique de la variance.....	101
Figure 41: Graphique des effets principaux sur la durée de résolution.....	102

Index des tableaux

Tableau 1: Description des scénarios.....	57
Tableau 2: Valeur des indicateurs pour les différents scénarios.....	61
Tableau 3: Résultats de la méthode ILS pour le scénario 1.....	62
Tableau 4: Résultats de la recherche d'atteignabilité pour le scénario 1.....	62
Tableau 5: Résultats de la recherche d'atteignabilité pour le scénario 1.....	63
Tableau 6: Résultats de la méthode ILS pour le scénario 2.....	63
Tableau 7: Résultats de la recherche d'atteignabilité pour le scénario 2.....	64
Tableau 8: Résultats de la méthode ILS pour le scénario 3.....	65
Tableau 9: Résultats de la recherche d'atteignabilité pour le scénario 3.....	65
Tableau 10: Résultats de la recherche d'atteignabilité pour le scénario 4.....	66
Tableau 11: Extrait du résultat de l'analyse fonctionnelle.....	75
Tableau 12: Résultats de la recherche d'atteignabilité pour le premier cas d'étude.....	93
Tableau 13: Données des cas de l'analyse pour le plan d'expériences.....	98
Tableau 14: Description des facteurs étudiés.....	99
Tableau 15: Les marges des postes.....	102

Introduction générale

Cette Thèse est le fruit d'une collaboration CIFRE (Convention Industrielle de Formation par la REcherche) entre l'entreprise TRANE et le Centre de Recherche en Automatique de Nancy (CRAN), unité de recherche mixte du CNRS et de l'université de Lorraine. L'entreprise TRANE est une entreprise internationale qui fabrique et commercialise une très large gamme de produits destinés à la climatisation et la réfrigération ainsi que des produits de conditionnement de l'air. La variété des produits au niveau de la puissance délivrée lui permet de proposer ses systèmes aussi bien aux particuliers qu'aux professionnels.

Pour répondre au mieux aux demandes spécifiques de ses clients, cette société, appartenant au groupe américain Ingersoll Rand, applique une philosophie nommée « Juste à Temps » (JiT) (Hall, 1987). Cette philosophie est appliquée dans les sites de production de l'entreprise par le biais de la mise en œuvre de la DFT (Demand Flow Technology) (Costanza, 1996) et du « lean manufacturing » dont le but est de satisfaire la demande client tout en réduisant les gaspillages tels que les tâches à non-valeur ajoutée, les stocks intermédiaires inutiles. Les systèmes de production sont organisés en lignes d'assemblage alimentées en sous-composants par d'autres lignes d'assemblage ou des cellules de fabrication (qui peuvent être communes à plusieurs lignes) et que l'on nomme « feeders ». Afin de limiter le stock de produits finis et de suivre au plus près la demande client, une cadence de sortie de ceux-ci est calculée comme étant le temps de production disponible pour une certaine période de temps (souvent la journée) divisé par le nombre de produits à assembler dans cette même période pour satisfaire les clients. La valeur obtenue est alors un objectif à atteindre et est nommée le « takt time ». Afin de limiter les en-cours, les produits à assembler vont passer d'un poste à l'autre de manière synchrone (au rythme de poste le plus lent). Le « takt time » devient alors la cadence à atteindre sur l'ensemble de la ligne.

Pour assurer cette synchronisation, la durée que met chaque poste pour réaliser sa charge de travail doit être la plus proche possible du « takt time » (et donc des autres postes de la ligne). On dit alors que la ligne est « équilibrée ». Pour un takt donné, lors de la conception de la ligne, le groupe « méthodes » de l'entreprise définit le nombre de postes, la répartition des tâches et des opérateurs et le séquençement des produits qui permettra d'atteindre ces objectifs. Ils réalisent donc l'équilibrage initial de la ligne.

Cependant, la ligne d'assemblage étant soumise à des aléas, parfois cet équilibrage initial ne pourra pas absorber tous les retards (la robustesse de l'équilibrage n'est donc pas parfaite). Il devient alors nécessaire, lorsque ce problème est détecté, de réagir afin d'ajuster au plus vite l'organisation de la production. Les produits assemblés sur la ligne d'assemblage considérée par notre étude étant de dimensions trop importantes (un produit peut atteindre 12 m de long), le re-séquencement n'est pas souhaité. De plus, l'appel à des opérateurs supplémentaires ne se fera qu'en dernier recours. Le but est donc, une fois l'aléa détecté, de trouver une nouvelle allocation des tâches sur les différents postes de la ligne puis de l'appliquer afin de continuer la production dans de bonnes conditions.

Résoudre ce problème de réallocation dynamique des tâches afin de rééquilibrer la ligne d'assemblage de manière dynamique est donc l'objet de cette thèse.
--

Au sein du laboratoire, et plus particulièrement au sein du département ISET, (Ingénierie des Systèmes Éco-Techniques complexes), les chercheurs se sont intéressés au problème de la synchronisation des flux d'informations et des flux matières (El-Haouzi, 2008 ; Thomas et al., 2009). Le problème de réallocation des tâches est lié à cette problématique car pour appliquer la solution trouvée, le système d'information doit être performant. Les études réalisées par les membres du département sur le concept de produit intelligent et sur les systèmes distribués peut trouver ici une application pour la détection de l'aléa puis par la diffusion de l'information pour l'application de la réallocation trouvée. Au niveau de la résolution proprement dite, des travaux précédents ont étudié la recherche d'atteignabilité sur un réseau d'automates communicants (dans le domaine plus large des Systèmes à Événements Discrets (SED)) pour résoudre des problèmes de séquençement (Marangé et al., 2011) ainsi que des problèmes d'allocation de tâches sur des contrôleurs. Ces problèmes sont similaires à notre problème de rééquilibrage dynamique : ce sont des problèmes de réallocation.

Nos travaux s'inscrivent donc à l'interface entre ces deux actions de recherche de notre laboratoire : l'amélioration du système d'information de l'entreprise et la résolution du problème d'allocation. La première partie, réalisée au sein de l'entreprise a permis de poser les hypothèses de détection des aléas et d'application des solutions trouvées. Ainsi la problématique scientifique se concentre sur la modélisation et la résolution des problèmes de rééquilibrage dynamique de lignes d'assemblage.

Le premier chapitre présente donc la problématique industrielle. Après avoir décrit le problème, les premiers travaux sont décrits. Ceux-ci permettent d'envisager la détection des aléas et d'appliquer les solutions trouvées permettant ainsi de concentrer la problématique sur le problème de la résolution d'un rééquilibrage dynamique de lignes d'assemblage.

Après avoir réalisé le parallèle entre notre problématique et les problèmes connus dans la littérature sous le nom de « problèmes d'équilibrage de ligne d'assemblage » (Assembly Line Balancing Problem, ALBP), un état de l'art des méthodes de résolution de ce type de problème est proposé dans le deuxième chapitre.

Cette étude de la littérature est complétée par une **première contribution** : l'application à notre cas d'étude de deux méthodes de résolution qui n'ont pas encore été utilisées dans ce domaine et dont les résultats sont comparés : la méta-heuristique « Iterated Local Search » (ILS) et une approche issue des SED. Dans celle-ci, chaque tâche et chaque poste de travail sont représentés par un automate communicant, le tout formant un réseau d'automates représentant le mécanisme d'allocation. Une recherche d'atteignabilité peut être alors effectuée sur ce réseau afin d'obtenir l'allocation correspondant à la solution recherchée.

Cependant, cette étude révèle une difficulté de modélisation et un manque d'adaptabilité aux différentes situations des usines de l'entreprise. En effet, si l'on souhaite modifier la façon dont on peut déplacer les tâches d'un poste à l'autre (leviers d'action), dans les deux méthodes étudiées, il faut redéfinir le jeu de contraintes et réinitialiser le modèle. Ainsi, si le problème de trouver une solution satisfaisante de rééquilibrage est soluble avec les deux approches, celui de la modélisation pour tenir en compte les différents scénarios de l'entreprise est encore un verrou scientifique.

Le troisième chapitre présente donc la **deuxième proposition** principale de la thèse qui est une méthode de modélisation et son application pour résoudre les problèmes de rééquilibrage dynamique de lignes d'assemblage. Celle-ci est basée sur une analyse fonctionnelle de la classe de problèmes afin d'obtenir un ensemble de modules. Ensuite, la méthode décrit comment ces modules sont implémentés sous la forme d'automates communicants pour modéliser chaque problème à résoudre. Enfin, la solution est obtenue par une recherche d'atteignabilité sur ce réseau d'automates communicants. En effet, le graphe d'états du modèle contient toutes les allocations de tâches possibles et la recherche d'atteignabilité permet de trouver une solution respectant l'objectif de « takt time » et d'ainsi nous fournir le moyen de la réaliser, c'est à dire répondre à la question de quelles tâches faut-il réallouer ?

Afin de valider cette proposition, le quatrième chapitre présente des applications de la méthode à différents cas d'étude. Nous verrons alors que malgré ses qualités de modélisation et de résolution, celle-ci connaît encore certains défauts, dont le risque d'explosion combinatoire. Cette application étudiera aussi l'impact des travaux réalisés pour l'entreprise et nous verrons l'impact sur les performances de l'absence de système de détection d'aléas ou d'application de la solution trouvée (validation des hypothèses).

Nous concluons donc ce mémoire sur l'intégration de notre méthode de résolution au sein du système plus complexe que constitue l'ensemble de la ligne d'assemblage. Nous mettrons en évidence aussi que l'utilisation d'algorithmes de recherche plus intelligents peut réduire le risque d'explosion combinatoire. Nous verrons aussi que l'amélioration de la cohabitation entre le « prédictif » et le « réactif » est une piste envisagée pour de futurs travaux.

Chapitre 1 : Contextes industriel et scientifique

Table des matières

I. Introduction.....	17
II. Objet d'étude : système de production & DFT.....	17
II.1. La DFT.....	17
II.2. Le takt time, la « voix du client » et outil de pilotage.....	18
<i>II.2.1. Conception des lignes d'assemblage.....</i>	<i>19</i>
<i>II.2.2. Équilibrage des lignes d'assemblage.....</i>	<i>19</i>
III. Problématique industrielle : le réactif en support du prédictif.....	21
III.1. Les variations à moyen ou/et long termes :.....	21
<i>III.1.1. Description.....</i>	<i>21</i>
<i>III.1.2. Leviers d'action.....</i>	<i>22</i>
III.2. Les variations à court terme:.....	23
<i>III.2.1. Description.....</i>	<i>23</i>
<i>III.2.2. Leviers d'action.....</i>	<i>24</i>
III.3. Proposition d'un équilibrage prédictif et réactif.....	25
IV. Contexte de recherche.....	26
IV.1. Les différents type de problèmes ALB.....	26
<i>IV.1.1. Introduction.....</i>	<i>26</i>
<i>IV.1.2. Définitions des problèmes d'équilibrage des lignes d'assemblage.....</i>	<i>27</i>
<i>IV.1.3. Classifications des SALBP.....</i>	<i>28</i>
<i>IV.1.4. Classifications des GALBP.....</i>	<i>28</i>
<i>IV.1.5. La reconfiguration ALBP.....</i>	<i>29</i>
IV.2. Problème scientifique.....	30
IV.3. Formalisation.....	31
<i>IV.3.1. Éléments du problème.....</i>	<i>31</i>
<i>IV.3.2. La prise en compte de l'aléa.....</i>	<i>31</i>
<i>IV.3.3. Objectif.....</i>	<i>32</i>
<i>IV.3.4. Contraintes d'allocation.....</i>	<i>32</i>
<i>IV.3.5. Contraintes de précedence.....</i>	<i>32</i>

<i>IV.3.6. Contraintes de parallélisation.....</i>	<i>33</i>
--	-----------

<u>V. Conclusion.....</u>	<u>33</u>
----------------------------------	------------------

I. Introduction

Ce chapitre a pour premier objectif de présenter le contexte industriel qui a motivé nos travaux de thèse ainsi que la problématique scientifique qui en découle.

Le problème industriel abordé concerne l'équilibrage de ligne pour des systèmes de production suivant une organisation en DFT (Demand Flow Technology). Dans le cadre industriel de l'entreprise TRANE, où cette étude a été menée, cette organisation est justifiée par le déploiement de lignes de production dédiées dans un contexte de personnalisation de masse. Nous commençons donc par décrire les principales caractéristiques d'un système de production en DFT et les problèmes posés par ce type d'organisation. En particulier, nous mettons en évidence le problème de la synchronisation entre les différents postes de travail constituant les lignes d'assemblage.

Parmi les solutions envisageables pour résoudre ce problème, des solutions issues du domaine du Lean Manufacturing (comme par exemple, l'implication des opérateurs dans la prise de décision) ou du domaine de recherche relatif aux produits intelligents avec les travaux d'El-Haouzi (El-Haouzi, 2008) peuvent être mises en place. Même si ces approches demeurent insuffisantes liées au manque d'information temps-réel sur les états de production, elles apportent des améliorations du système d'information qui ont permis de mieux synchroniser les lignes d'assemblage entre elles. C'est sur cette base, que sera posé le problème de rééquilibrage dynamique entre les postes au sein d'une même ligne d'assemblage.

La dernière partie du chapitre sera consacrée à la formalisation du problème industriel posé.

II. Objet d'étude : système de production & DFT

II.1. La DFT

Dans ce mémoire nous nous intéressons à des systèmes de production organisés selon les préceptes de la DFT (Demand Flow Technology). Cette méthode a été initiée par (Costanza, 1996) et tient ses origines du système de production Toyota comme plusieurs approches telles que la théorie des contraintes, la production synchrone, le Lean pour n'en nommer que quelques-unes. Ces approches ont été créées pour répondre à un enjeu industriel majeur qui est la personnalisation de masse (mass customization). Cette méthode de management de la demande est caractérisée par une variabilité toujours croissante des produits et des volumes de production.

La DFT s'appuie sur plusieurs étapes standardisées et des outils mathématiques pour répondre aux objectifs suivants :

- La suppression des tâches à non valeur ajoutée (dans un objectif d'amélioration continue toutes les tâches à valeur ajoutée ou non sont listées et caractérisées) ;
- La réduction des coûts par la standardisation des processus ;
- L'augmentation de la qualité afin de répondre aux exigences clients (délais et volume) ;
- L'augmentation de la flexibilité par l'utilisation de ressources flexibles (opérateurs travaillant sur plusieurs postes, des lignes dédiées et reconfigurées pour suivre la demande) ;

- L'optimisation du pilotage des flux en adaptant une gestion en flux tirés ;

Pour répondre à ces objectifs, la DFT entraîne une réorganisation de l'espace productif en lignes de production spécifiques, en pipelines. Nous considérerons ainsi, qu'au centre de notre usine, existe un flux principal de production. Pour alimenter ce flux, divers sous-ensembles sont nécessaires. Ceux-ci sont fabriqués sur des lignes d'assemblage appelées « feeders » ou sur des cellules indépendantes.

Pour résumer la DFT impose une standardisation stricte des procédés et des méthodes de travail dans l'objectif de répondre aux exigences des clients : on parle alors de prendre en compte la « voix du client ».

II.2. Le takt time, la « voix du client » et outil de pilotage.

Pour atteindre son objectif de flexibilité, de qualité et de réduction des tâches à non-valeur ajoutée, la DFT s'appuie sur une démarche en 20 étapes. Le pilotage de la production étant guidé par les besoins des clients, l'une des premières étapes est l'estimation de la demande client (nombre, type de produits, délais de livraison, prix, etc.). Une fois celle-ci évaluée, afin de la respecter au mieux, la production s'appuie sur certains indicateurs. Le premier de ceux-ci est l'estimation du nombre de produits qu'il faut assembler par jour afin de satisfaire tous les clients de l'entreprise. Celui-ci est noté DC, pour « demand at capacity », et est évalué régulièrement (tous les 3 mois dans l'entreprise Trane) afin de tenir compte des demandes effectives des clients et des prévisions de vente réactualisées.

À partir de cet indicateur et avec la connaissance du nombre d'heures de production effective de la journée (temps d'ouverture corrigé pour tenir compte de l'efficacité et de l'efficience de la ligne d'assemblage), il est possible d'en déduire la cadence de sortie des produits finis nécessaire pour satisfaire cette demande journalière, tout en respectant les exigences en matière de quantité et de délai.

Cet objectif de rythme de sortie des produits au dernier poste de la ligne d'assemblage est alors appelé le « takt time ». Par exemple, si l'entreprise dispose de 8 heures de travail dans la journée et qu'elle doit assembler 10 produits, alors il faudra qu'un produit soit terminé toutes les 48 minutes (0,8 heures).

Ainsi la formulation du takt time est exprimée par :

$$takt = \frac{TP}{DC} \text{ avec TP le nombre d'heures de production effective, le temps de production (inférieur au temps d'ouverture car dépendant de l'efficacité et de l'efficience) et DC le nombre de produits}$$

Ce « takt time » est alors l'objectif à atteindre pour que la cadence de sortie des produits satisfasse l'ensemble des clients. Cependant, le dernier poste de travail de la ligne ne peut pas le respecter seul. Comme nous le verrons dans la partie suivante, le respect du « takt time » devient un des objectifs de l'ensemble de la ligne (et des fournisseurs internes).

Ce takt est alors un chef d'orchestre qui pilote toute l'usine.

Les lignes d'assemblages sont donc conçues de manière à respecter le rythme imposé par cette cadence (ce battement de tambour) ; c'est ce que nous allons décrire dans la partie suivante.

II.2.1. Conception des lignes d'assemblage

Chaque ligne d'assemblage est une succession de postes sur lesquels les produits passent (Figure 1). Afin de réduire les en-cours, il n'y a pas de stocks intermédiaires entre les postes : les produits doivent donc passer d'un poste à l'autre de manière synchronisée. Le déplacement de tous les produits sera donc conditionné par le poste le plus lent dont la durée définira la cadence de la ligne de production. Ainsi, tous les postes doivent aussi avoir une durée de travail similaire correspondant au takt : c'est ce qu'on appelle l'équilibrage de ligne d'assemblage.

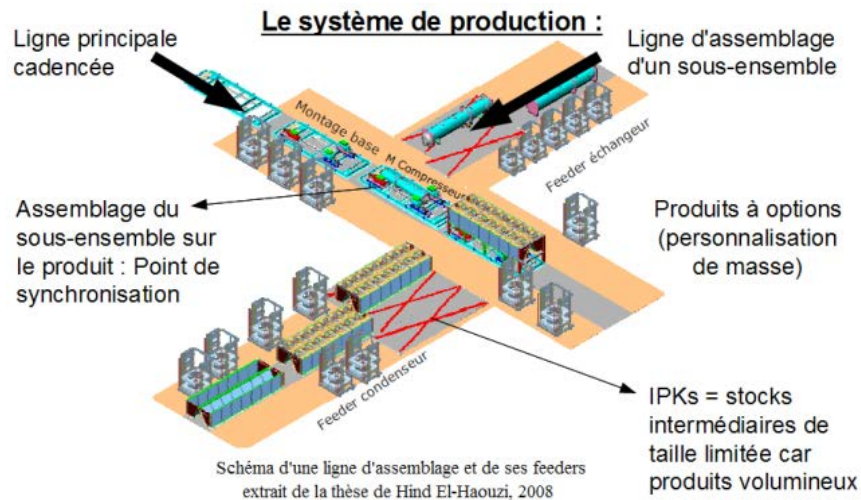


Figure 1: Ligne de production et ses feeders.

Chaque ligne d'assemblage est alimentée par le magasin ou par des lignes d'assemblage de sous-composants que l'on nomme « feeders ». La synchronisation entre les feeders et les lignes est compliquée car les sous-composants peuvent être communs à plusieurs lignes et en nombres différents, générant ainsi des cadences de production différentes. Ceci est alors géré par des IPK (In Process Kanban). Un IPK est un emplacement où les unités stationnent entre deux postes d'assemblage (un stock), ce qui permet d'assurer la synchronisation des feeders avec la ligne principale. Ce type de kanban est matérialisé au sol par une croix barrant un rectangle à la dimension du produit fabriqué. Cela permet de découpler la ligne et ses feeders, ils possèdent donc chacun leur takt time et leur propre équilibrage.

II.2.2. Équilibrage des lignes d'assemblage.

a. L'équilibrage selon la DFT

L'ensemble des opérations à réaliser pour obtenir un produit fini est constitué de tâches élémentaires indivisibles qui sont ensuite réparties en opérations le long des postes de la ligne, pour être réalisées par un ou plusieurs opérateurs. Chaque tâche étant alors décrite dans les SOE (Sequence Of Events) qui formalisent la durée de chacune de celles-ci ainsi que leurs enchaînements. Dans le détail d'une tâche se trouve le temps entre deux actions à valeur ajoutée, actions à non-valeur ajoutée, la durée du contrôle des produits, de la maintenance de premier niveau. Mais on trouve aussi le nombre d'opérateurs nécessaires, les outils qui doivent être utilisés, etc.

Lors de la conception de la ligne, la répartition des tâches est réalisée afin que le « takt time » soit respecté sur l'ensemble des postes de travail.

Afin d'être réalisable, cette répartition doit prendre en compte différentes contraintes :

- Contraintes d'allocation : certaines tâches nécessitent un outillage particulier : celui-ci doit donc être disponible sur les postes de travail où l'on souhaite les allouer.
- Contraintes de précédence : certaines tâches doivent être terminées avant que d'autres ne soient commencées. Ces contraintes sont souvent représentées dans les gammes logiques par des diagrammes de précédence où chaque arc du schéma représente une contrainte de prédécesseur/successeur.
- Contraintes de parallélisation : certaines tâches peuvent être réalisées simultanément par deux opérateurs différents qui ne doivent pas se gêner dans leur travail respectif. On dit alors qu'elles sont effectuées en parallèle. Cela est possible grâce à la grande dimension des produits.

Ces contraintes sont initialement définies par le processus de fabrication et par la définition d'une gamme logique du produit listant tous les processus. Bien que la littérature, propose des algorithmes permettant de trouver l'équilibrage initial, dans la pratique beaucoup d'entreprises n'appliquent pas ces méthodes seules. En effet, l'entreprise préfère impliquer ses différents services afin de s'assurer que l'équilibrage qui sera appliqué sera bien accepté par tous. De plus, la cohabitation de différents corps de métier permet d'identifier, lors de l'établissement des compromis, différentes contraintes ou améliorations qui n'avaient pas été formalisées précédemment. Ainsi, lors de ces discussions, les opérateurs peuvent par exemple signaler la difficulté de réaliser certaines tâches, ou bien certaines tâches à non valeur ajoutée peuvent être identifiées et réduites (voire éliminées). Cette collaboration est une des philosophies du LEAN permettant que la décision finale soit facilement acceptable et ne génère pas le refus que pourrait causer une solution qui serait imposée par une application pure de la DFT ou des algorithmes mathématiques. (Évidemment, cela n'est pas incompatible avec un système d'aide à la décision qui peut apporter des suggestions). Le travail consiste ensuite à améliorer le lissage en déplaçant des tâches, en ajoutant des opérateurs là où cela est nécessaire ou encore en réduisant la durée de certaines tâches.

Pour que ces différentes équipes travaillent ensemble et trouvent le meilleur compromis (entre l'équilibrage et les contraintes imposées), elles ont besoin d'échanger des informations claires. Ces informations incluent notamment la représentation de l'équilibrage.

b. Représentation de l'équilibrage

L'équilibrage initial est souvent représenté dans l'entreprise par un « diagramme Yamazumi ». Ce diagramme est un des outils utilisés dans le cadre d'une politique de « LEAN manufacturing ».

Celui-ci consiste à représenter chaque tâche à effectuer sur le produit par un bloc dont la hauteur est proportionnelle à la durée de la tâche. Ces blocs sont ensuite empilés en plusieurs colonnes correspondant à chaque poste de travail en fonction des opérateurs disponibles.

Ainsi, l'équilibrage de la ligne d'assemblage est représenté de manière graphique et les postes ne respectant pas le takt time sont identifiés visuellement (Figure 2).

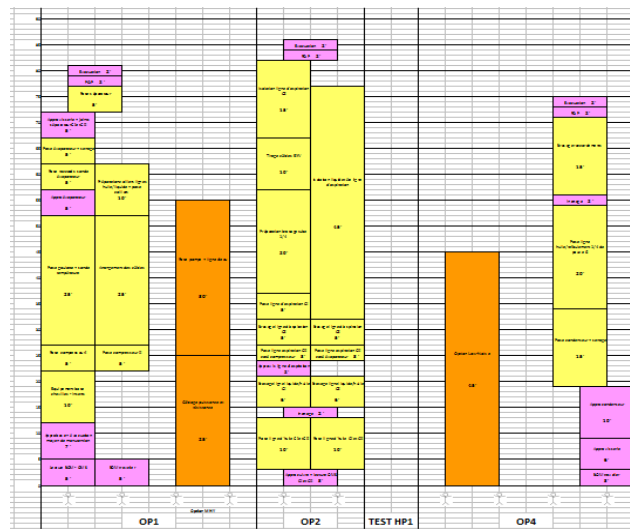


Figure 2: Diagramme Yamazumi représentant un équilibrage initial.

Nous avons vu que toute l'organisation et le pilotage d'une entreprise mettant en place une organisation de type DFT sont basés sur le takt, or le takt peut être soumis à des variations, quelles soient au niveau de la « DC » (demande client) ou au niveau du temps de production des opérateurs « TP». Par ailleurs d'autres variations ou perturbations peuvent s'ajouter, comme par exemple, les perturbations dues à des aléas du terrain, telles des pannes ou tout simplement les variations naturelles de l'outil de production.

Tout cela implique un déséquilibre des lignes de production et donc un besoin fort de rééquilibrage de ces lignes.

III. Problématique industrielle : le réactif en support du prédictif

Pour comprendre les causes des variations et donc en dégager des leviers d'action, nous séparons les variations en deux catégories en fonction de l'horizon temporel : les variations à long ou/et moyen termes et les variations à court terme.

III.1. Les variations à moyen ou/et long termes :

III.1.1. Description

Les variations moyen ou long termes ont la caractéristique de pouvoir être en partie prévisibles. Le système de production peut donc les anticiper. Elles peuvent être dues à plusieurs facteurs :

- Les processus de production peuvent évoluer, changeant ainsi la durée des tâches. Cela peut être dû au phénomène d'apprentissage des opérateurs, ou à la modification d'une machine de production...
- Le nombre d'opérateurs disponibles change au cours du temps, mais il est en partie prévisible à travers l'analyse du planning.

- La demande client n'est pas constante. Une équipe de l'entreprise est dédiée à la prévision de celle-ci et grâce à cela elle (re-)définit les plannings de production.

III.1.2. Leviers d'action

Pour limiter l'impact négatif de toutes ces variations, on trouve dans la littérature, deux approches possibles (Billaut et al., 2005) : proposer une solution robuste permettant ainsi d'absorber les différents aléas ou proposer un système flexible qui adapte la solution lorsqu'un aléa survient. La DFT, par exemple, propose un ajustement de la DC (demand at capacity) chaque 3 ou 6 mois et propose de refaire tout l'équilibrage de la ligne chaque 3 ans. De manière plus précise, les usines adaptent leur production de la semaine pour tenir compte des disponibilités et des compétences des opérateurs. Ce délai d'une semaine permet de tenir compte des congés et des maladies tout en assurant un délai réglementaire pour prévenir un opérateur dont les horaires de travail seraient modifiés.

Ces différents niveaux de recalculs apparaissent aussi dans la littérature. En effet, (Scholl, 1999) identifie 3 horizons temporels pour les équilibrages de lignes d'assemblage : de 2 à 5 ans lors de l'installation d'une nouvelle ligne d'assemblage, de 6 mois à 2 ans pour une réingénierie et une reconfiguration de la ligne (Grangeon et al., 2011 ; Makssoud et al., 2014) et enfin de 1 mois à 1 an pour un rééquilibrage pour adapter la production à un changement du nombre de demandes clients.

Concernant les variations à moyen terme et donc d'horizon plus faible (inférieur au mois), la modification du séquençement est privilégiée. Ainsi l'équilibrage des lignes d'assemblage d'automobiles est réalisé de façon à pouvoir absorber les retards en modifiant le séquençement sur la ligne d'assemblage (appelé film, ou « car sequence ») (Lesert et al., 2011).

Afin de prendre en compte ces variations prévisibles et d'éviter le processus de recalcul complet de l'équilibrage à chaque changement des conditions de production (celui-ci pouvant être long et fastidieux), les concepteurs de ligne d'assemblage peuvent prévoir différents scénarios qui seront ensuite appliqués en fonction des choix et des besoins de l'entreprise. Pour l'entreprise Trane, cet ensemble de scénario est dénommé le takt playbook (traduction littérale : notice de takt) et un exemple est présenté sur la Figure 3. Cette anticipation permet ainsi de disposer d'autres équilibrages qui seront appliqués lorsque les paramètres de production auront évolué en-dehors des limites de robustesse de l'équilibrage initial qui sont défini par le takt $\pm \varepsilon$ (cette tolérance étant initialement définie).

TAKT PLAYBOOK LIGNE RTAF										
Nb unités par jour	TAKT TIME	P0	P1	P2	P3	P4	P5	P6	P7	TOTAL OPERATEURS
2	210 Min	 				POSTE FERME				8
3	140 Min	 								10
4	105 Min		 							13
5	84 Min		 	 						16

Figure 3: Exemple d'un "Takt playbook" simplifié.

Ainsi, au début de la semaine, le gestionnaire de ligne prévoit le nombre d'opérateurs disponibles, et en fonction du nombre de produits qui doivent être assemblés, il peut définir la meilleure configuration de la ligne d'assemblage pour la semaine. Cette configuration inclut la valeur du takt time, le nombre de postes en service et le nombre d'opérateurs sur chacun d'eux.

Par conséquent, lorsque la demande client évolue (par exemple pour suivre une saisonnalité) ou lorsque le nombre d'équipes varie (par exemple pour les congés du personnel), l'équilibrage initial est déjà calculé et n'a plus qu'à être appliqué dans l'usine. Celui-ci devient alors la norme jusqu'à la prochaine reconfiguration.

Pour autant, d'autres causes peuvent nécessiter de modifier le takt et il faudra les identifier. Pour cela il faut mettre en place des indicateurs.

Le 1^{er} challenge :

Quelles sont les autres causes qui nécessitent de reprendre un nouveau takt ?

Quel indicateur mettre en place pour basculer d'un équilibrage à un autre ?

III.2. Les variations à court terme:

III.2.1. Description

Une fois la ligne conçue et sa configuration initiale définie, les équipes de travail sont constituées et la production lancée. Cependant, les lignes d'assemblage subissent des aléas qui génèrent des retards dont certains, trop importants, ne sont pas absorbés naturellement par le séquençement des différents produits ou par la flexibilité des opérateurs eux-mêmes.

Les aléas peuvent être les suivants :

- Les durées de chaque tâche peuvent changer :

- Le temps machine n'est pas strictement le même d'un produit à l'autre : nous sommes face à un phénomène aléatoire, quelques fois même avec une déviation de type « systématique » comme celle due à l'usure des outils.
- Les durées pour réaliser une tâche dépendent des opérateurs : par exemple, un opérateur fatigué ou inexpérimenté sera plus lent.
- Un aléa de production peut survenir :
 - une panne de l'outillage, dans ce cas la durée d'une ou plusieurs tâches est modifiée (retard, ou bien passage sur une autre machine, utilisation d'un autre outil moins adapté)
 - une rupture d'approvisionnement : dans ce cas, le retard est dû à un temps d'attente de la pièce manquante. Certaines opérations peuvent être réalisées, mais elles ne doivent pas gêner l'assemblage de la pièce manquante quand elle arrivera (contrainte de précédence), sinon il faudra en démonter une partie (retard par ajout de temps de démontage)

Lorsque les aléas apparaissent, ils retardent le produit en cours de transformation au poste concerné. Afin de minimiser l'impact sur la production, et donc en diminuer leurs coûts, une action corrective doit être rapidement mise en place.

III.2.2. Leviers d'action

Dans la littérature au niveau du pilotage court terme deux leviers d'action sont souvent étudiés : la modification du séquençement de produits ou la flexibilité des opérateurs (ré-assignation des opérateurs).

a. Modification du séquençement :

La répartition des tâches est souvent couplée à la définition de règles de séquençement contrôlant la succession des produits sur la ligne d'assemblage (« mixed model » car les produits passant sur une ligne sont de la même famille, mais disposent d'options différentes) (exemple : un produit de grandes dimensions doit être suivi de deux produits de dimensions restreintes) (Araoui et al., 2013). Dans certaines usines, le reséquencement n'est pas modifié. Cela peut être dû aux grandes dimensions des produits, être contraint par la livraison de matière sur chaque poste d'assemblage, etc.

b. Flexibilité des opérateurs :

Si un opérateur termine son travail avant la fin du takt, celui-ci peut débiter une autre tâche. S'il dispose des compétences nécessaires, il peut changer de poste de travail pour travailler sur un autre produit. Ce travail peut être effectué seul, mais si certaines tâches le permettent, il peut être effectué en parallèle d'un autre collègue (étant en retard ou non). L'opérateur supplémentaire peut aussi venir d'une autre ligne, mais afin de limiter les déplacements, les flex se limitent si possible aux deux postes voisins du poste de l'opérateur (en amont et en aval). Le flex peut aussi être favorisé dans un sens (vers l'aval de la ligne), ou pour certaines compétences rares parmi les opérateurs (Zavadlav et al., 1996). C'est pourquoi la position des opérateurs avec les bonnes compétences doit donc être étudiée dès la conception de la ligne.

c. Nouvelle répartition des tâches, le rééquilibrage :

Ce levier est très peu étudié dans la littérature en particulier dans le cadre d'aléas ou d'événements à court terme. Ces aléas peuvent être ponctuels ou impacter les produits futurs. C'est par exemple le cas pour une panne : la machine concernée fonctionnera en mode dégradé tant qu'elle n'aura pas été réparée. L'action corrective devra alors prendre en compte cela afin de calculer une configuration prédictive pour les futurs produits (tout en restant temporaire et ne concernant que les produits considérés).

La survenue régulière d'aléas peut indiquer un problème dans l'équilibrage initial. Cela peut être dû à de mauvaises informations initiales, à un changement du processus de fabrication, etc. Dans tous les cas, cela indique que l'équilibre initial n'est plus adapté : celui-ci doit donc être recalculé entièrement en analysant les causes de celui-ci afin d'éviter qu'il ne se reproduise.

Le 2^{ème} challenge :

Suite à un événement significatif impromptu, proposer un nouvel équilibrage dans un laps de temps très court (la détection du retard et l'application de la solution étant supposés facilement réalisables).

Pour répondre aux deux challenges industriels et pour mieux gérer les différentes variabilités auxquelles les usines font face, un système de pilotage prédictif /réactif pourra alors être proposé.

III.3. Proposition d'un équilibrage prédictif et réactif

Nous allons voir dans cette partie un scénario d'usage mettant en place un pilotage hybride (prédictif/réactif) afin d'assurer la gestion des aléas au sein du système de production. Comme nous l'avons décrit dans les parties précédentes, la première étape (celle réalisée avant même le début de la production), consiste à calculer plusieurs équilibrages initiaux de la ligne et de les compiler dans le « takt playbook ».

Le gestionnaire de la ligne choisira ensuite une de ces configurations en fonction de la demande client de la semaine et des opérateurs disponibles dans son équipe, puis la production pourra être lancée.

Pendant cette production, un aléa peut apparaître. Lorsque celui-ci est détecté, et s'il n'est pas trop tard, une mesure corrective est cherchée puis appliquée. L'événement peut être alors enregistré en mémoire. Si le retard est détecté trop tard, une autre mesure corrective sera appliquée, mais il sera alors plus difficile de corriger les effets indésirables. L'événement devra tout de même être sauvegardé.

Si l'aléa impacte plusieurs produits et ce, pendant un temps assez long ou si des retards surviennent trop souvent, la reconfiguration devra tenir compte des éléments prédits (comme la durée estimée du retard) afin d'être plus précise pour limiter les conséquences d'une quelconque dérive. Cependant, cet horizon temporel plus long permet de disposer d'un peu plus de temps pour effectuer le calcul. Le rééquilibrage peut donc être calculé de manière prédictive et globale.

Si le système de contrôle des événements détecte la répétition d'un aléa, ce recalcul peut même devenir définitif et demander un recalcul de tous les équilibrages contenus dans le takt playbook en tenant compte des différences entre les paramètres précédents et leurs valeurs actuelles qui génèrent cet aléa répétitif (modification du processus de fabrication, etc.).

Pour détecter cela, le gestionnaire de ligne utilisera alors les enregistrements des événements afin de suivre les aléas et leurs causes afin d'identifier celles revenant souvent.

Ces étapes décrivent l'interaction entre l'approche prédictive et l'approche réactive et entre les différents niveaux de prise de décision. Elles peuvent être résumées sur le logigramme de la figure ci-dessous (Figure 4).

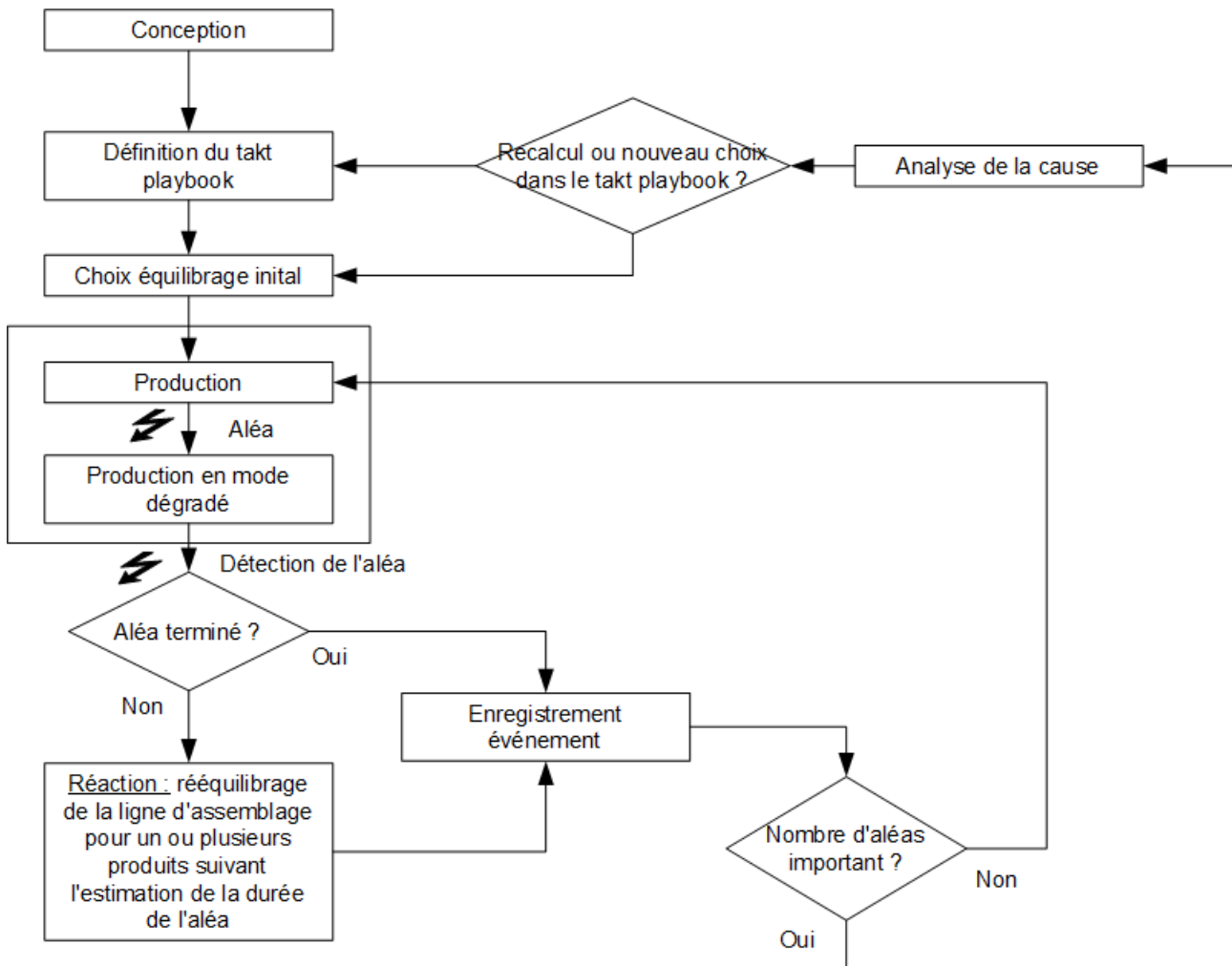


Figure 4: Interaction entre les calculs prédictifs et les aléas.

IV. Contexte de recherche

IV.1. Les différents type de problèmes ALB

IV.1.1. Introduction

Les problèmes d'équilibrage de ligne d'assemblage ont intéressé la communauté scientifique et plus particulièrement celle de la recherche opérationnelle. Ceci dès l'apparition des premiers modèles d'équilibrage de ligne d'assemblage par (Salveson, 1955) et le besoin de trouver des lignes d'assemblages efficaces.

Les premiers états de l'art sont apparus en 1973 et 1986 (Buxey, 1973 ; Baybars, 1986 ; Talbot et al., 1986). Et cela continue d'intéresser la communauté actuelle qui propose toujours de nombreuses méthodes de résolution ainsi que des états de l'art plus récents et des classifications (Scholl, 1999 ; Becker and Scholl, 2006 ; Scholl and Becker, 2006 ; Boysen et al., 2007 ; Boysen et al., 2008 ; Battaïa and Dolgui, 2013).

L'arrivée de la personnalisation de masse a obligé les lignes à être repensées afin d'accueillir plusieurs produits sur une même ligne d'assemblage tout en limitant le nombre de produits stationnant dans la ligne. Pour prendre en compte cela, les modèles ont été modifiés et les chercheurs proposent de nouvelles méthodes de résolutions de plus en plus efficaces.

IV.1.2. Définitions des problèmes d'équilibrage des lignes d'assemblage

Comme nous l'avons décrit dans le problème industriel, une ligne d'assemblage est définie par un ensemble de postes de travail w_i (souvent placés de manière linéaire, afin de limiter les déplacements entre postes - si certains postes sont doublés, on précisera que ceux-ci sont en parallèles). Les postes peuvent être automatiques ou manuels.

Les produits passent séquentiellement de poste en poste jusqu'à sortir de la ligne sous forme de produits finis. Pour cela, des tâches élémentaires t_i doivent être réalisées sur le produit. Chaque tâche est allouée à un poste de travail.

À cela, il faut ajouter les contraintes de précédence qui imposent que certaines tâches soient terminées avant d'autres pour des raisons technologiques. (Par exemple, il faudra réaliser toutes les connections dans un coffret électrique avant de fermer le couvercle de celui-ci.) Ces contraintes sont souvent représentées sous la forme d'un graphe de précédence illustrant toutes les relations entre prédécesseurs et successeurs (Figure 5).

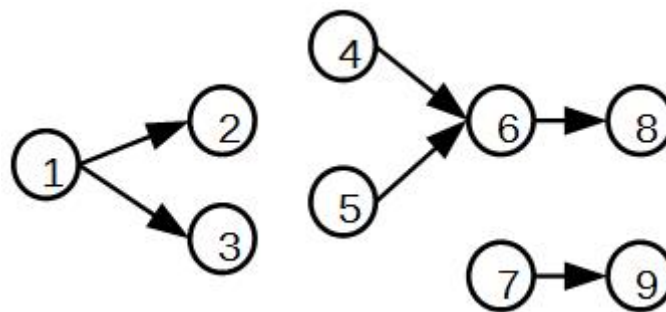


Figure 5: Exemple d'un graphe de précédence.

Le problème de l'équilibrage d'une ligne d'assemblage consiste à trouver l'allocation de chacune des tâches de façon à respecter ces contraintes tout en optimisant la répartition de la charge de travail, c'est à dire le lissage.

Que ce soit de manière manuelle ou par un convoyeur, le changement de poste de tous les produits se fait de manière synchrone. Ainsi, si la charge de travail est mal lissée, les postes de travail devront attendre le poste le plus lent. Cette durée est alors définie comme le temps de cycle, c'est à dire le temps que passe chaque produit à chaque poste de travail (celui-ci étant égal au minimum de la somme des temps de travail

de chaque poste). L'objectif est d'avoir ce temps de cycle égal au takt time défini dans le premier chapitre afin de n'être ni en avance, ni en retard.

IV.1.3. Classifications des SALBP

L'équilibrage défini ainsi est alors dit simple (SALBP, Simple Assembly Line Balancing Problem) (Baybars, 1986).

La prise en compte d'autres typologies de lignes, d'autres contraintes ou encore d'autres objectifs forment des problèmes dits généraux (GALBP, General Assembly Line Balancing Problem) dont nous parlerons dans la partie suivante.

Les éléments pris en compte ici sont les contraintes de précédence, le temps de cycle et le nombre de postes de travail afin de trouver la meilleure allocation avant l'installation de la ligne d'assemblage. Selon (Baybars, 1986) et (Scholl, 1999), il existe quatre types de SALBP se différenciant en fonction de l'objectif :

- Type 1 : le temps de cycle est fixé et l'objectif est de minimiser le nombre de postes de travail.
- Type 2 : le nombre de postes de travail est fixé et l'objectif est de minimiser le temps de cycle.
- Type E : l'objectif est de minimiser en parallèle le temps de cycle et le nombre de postes de travail.
- Type F : le temps de cycle et le nombre de postes de travail sont fixés et l'objectif est de savoir s'il existe une allocation possible, c'est un problème de vérification de faisabilité.

IV.1.4. Classifications des GALBP

Les problèmes généraux regroupent tous les problèmes qui ne sont pas simples : ils sont donc très variés.

Les différences par rapport au problème simple peuvent être :

- l'ajout de postes de travail en parallèle
- les lignes sont disposées en U (et donc avec des postes partageant des ressources)
- l'ajout de contraintes d'assignation (certaines tâches nécessitent certains outillages ou certaines compétences disponibles uniquement sur certains postes)
- la prise en compte de l'espace occupé au poste par la matière à fournir pour chaque tâche (contraintes de temps et d'espace, TSALBP)
- une famille de produits fabriqués sur la même ligne (multi-model ou mixed-model, par rapport à un seul produit assemblé en grande série)
- Explication de la différence entre multi-model et mixed-model (Becker & Scholl, 2006)
- etc.

Afin de mieux comparer les problèmes entre eux, la classification de ces problèmes est nécessaire. (Boysen et al., 2007) proposèrent un formalisme inspiré par les n-uplets utilisés par Graham en 1979 (Graham et al., 1979) pour classer les problèmes d'ordonnancement.

Ainsi, chaque problème d'équilibrage peut être traduit par un n-uplet $[\alpha \mid \beta \mid \gamma]$ où :

- α décrit les caractéristiques et les contraintes des tâches à réaliser sur les produits (et donc les caractéristiques du graphe de précédence)
- β décrit les caractéristiques et les contraintes de la ligne d'assemblage et des postes de travail qui la composent
- γ décrit les objectifs du problème

Cette classification détaillée dans l'article (Boysen et al., 2007), peut être retrouvée sur le site internet suivant : <http://alb.mansci.de/>. Sur celui-ci, un formulaire à remplir permet d'obtenir le n-uplet correspondant aux données du problème renseigné et de le comparer aux articles enregistrés dans la base de données.

Chaque composition forme alors un problème spécifique. Dans son article, « Assembly line balancing: Which model to use when? », (Boysen et al., 2008) propose un état de l'art des méthodes de résolution en lien avec cette classification.

Malgré l'abondance de la littérature dans le domaine, il subsiste encore des sujets de réflexion comme par exemple, la difficulté à identifier les problèmes. En effet, dans le monde réel les problèmes à traiter sont plus complexes car ce sont des combinaisons de plusieurs extensions simultanées du problème de base (Falkenauer, 2005 ; Boysen et al., 2008).

Chaque problème conduisant à une unique combinaison de facteurs, **comment trouver une méthode de résolution adaptée ?**

IV.1.5. La reconfiguration ALBP

Dans cette thèse, au vu du problème industriel posé, nous nous intéressons donc au problème de reconfiguration et à sa combinaison avec d'autres contraintes.

Les définitions des dictionnaires concernent principalement le domaine de l'informatique. Ainsi, pour le dictionnaire Larousse, une configuration informatique est : l'ensemble des caractéristiques matérielles et logicielles d'un système informatique (processeur, mémoire, système d'exploitation...) ou la modification ou réglage de paramètres informatiques en vue de l'optimisation du fonctionnement du système.

De même, selon l'office québécois de la langue française, une reconfiguration informatique correspond à une modification de la configuration des éléments matériels et logiciels d'un système informatique, dans le but d'accomplir une fonctionnalité différente ou de permettre la poursuite des opérations après la défaillance d'une unité.

Dans le cas des systèmes de production, (Koren et al., 1999) définissent ainsi un système manufacturier : « A Reconfigurable Manufacturing System (RMS) is designed at the outset for rapid change

in structure, as well as in hardware and software components, in order to quickly adjust production capacity and functionality within a part family in response to sudden changes in market or in regulatory requirements. »

L'ajout du mot dynamique permet de prendre en compte le besoin de rapidité de réponse et le fait que cela est fait pendant la production, et non pas à l'arrêt entre 2 périodes de production. En effet, certaines reconfigurations ne sont pas dynamiques : il n'y a pas de contraintes fortes de temps de réponse. La production continue, le système n'est pas arrêté.

De plus, cet aspect dynamique se manifeste aussi dans le nombre des reconfigurations. En effet, ici le système sera fortement dynamique car il sera amené à s'adapter souvent, alors que les réconceptions sont plus rares et n'ont lieu que lors de changement de production. Il y a en effet différents horizons de reconfiguration (Scholl, 1999).

Dans cette thèse nous définirons donc la reconfiguration comme le changement des éléments d'une ligne d'assemblage afin d'adapter celle-ci lors d'une modification de ses paramètres (changement d'objectif, aléa perturbant la production, etc.).

Ce besoin de reconfiguration des systèmes manufacturiers a été mis en lumière par (Koren et al., 1999) qui a défini les RMS (reconfigurable Manufacturing Systems) comme un intermédiaire entre les FMS (Flexible Manufacturing Systems) et les lignes dédiées pour réaliser à la fois de grandes quantités de produits avec une grande diversité. (Mehrabi et al., 2000) a insisté en annonçant que les RMS sont la clé des futurs systèmes de production.

Nous faisons l'hypothèse que les performances croissantes des systèmes d'information permettent de transférer efficacement aux opérateurs la nouvelle réallocation des tâches afin de nous concentrer sur la recherche rapide d'une solution.

Un autre manque relevé dans la littérature est que la majorité des articles traitent du problème d'équilibrage à la conception de la ligne d'assemblage avant une première installation. Très peu traitent de la reconfiguration de la ligne d'assemblage, et en particulier de **la reconfiguration ou du rééquilibrage dynamique face à un événement très court terme.**

IV.2. Problème scientifique

Le problème industriel présenté ci-dessus est donc un problème du type ALBP mais pour lequel on ajoute un élément dynamique et aléatoire: la prise en compte d'un aléa impactant une des tâches à réaliser et donc un des postes de la ligne.

Il s'agit d'allouer un ensemble de tâches T sur un ensemble de postes de travail W en respectant plusieurs contraintes. Il existe plusieurs allocations possibles qui peuvent être classées selon leur degré d'intérêt pour l'entreprise. Comme notre contexte est la reconfiguration, la vitesse de résolution est privilégiée, mais pour autant, une réallocation qui engendre le moins de perturbations est souhaitable. Après avoir étudié ces différentes classifications, nous en déduisons que notre problème est un problème de GALBP à cause de la reconfiguration et de la possibilité de paralléliser les tâches.

Nous allons ci-après décrire les éléments du problème (tâches et postes de travail), puis l'objectif de la reconfiguration et enfin les contraintes.

IV.3. Formalisation

IV.3.1. Éléments du problème

- $T = \{t(i) \in N, i \leq T_{max} \in N\}$ est l'ensemble des tâches (avec t_i l'identifiant de la tâche et T_{max} le nombre de tâches)
- $W = \{w(i) \in N, i \leq W_{max} \in N\}$ est l'ensemble des postes de travail de la ligne (avec w_i l'identifiant du poste et W_{max} le nombre de postes)
- $E = \{e(i) \in N, i \leq E_{max} \in N\}$ est l'ensemble des événements de synchronisation, c'est-à-dire les instants où les produits changent de postes sur la ligne (avec E_{max} le nombre d'événements pendant la période étudiée)
- $D_t = \{d_{t(i)} \in N, i \leq T_{max} \in N\}$ est l'ensemble des durées des tâches
- $D_w = \{d_{w(i)} \in N, i \leq W_{max} \in N\}$ est l'ensemble des durées des postes de travail qui sont définies par les sommes des durées des tâches qui sont allouées à chaque poste

IV.3.2. La prise en compte de l'aléa

L'aléa est modélisé par un retard qui, à un instant donné, impacte un seul poste de la ligne. Ce retard impacte donc un seul produit : celui actuellement présent sur le poste. Le rééquilibrage dynamique doit donc être trouvé et appliqué avant la fin du takt time, afin de ne pas impacter les autres produits assemblés sur les autres postes (ligne synchrone).

- $t(r)$ est alors la tâche pendant laquelle le retard est détecté.
- $D_{t(r)}$ est la durée du retard, celle-ci est ajoutée à la durée de la tâche impactée $t(r)$
- $T_{bis} = \{t(i) \in N, i \leq T_{max} \in N\}$ est alors l'ensemble des tâches non terminées qui peuvent être réallouées.
- $T - T_{bis}$, représente les tâches bloquées, elles ne pourront pas être déplacées. Cependant leur durée est prise en compte afin de ne pas surcharger le poste où elles sont réalisées. En effet, les tâches déjà terminées ne peuvent plus être déplacées.
- $W_{bis} = \{w(i) \in N, i \leq W_{max} \in N\}$ est alors l'ensemble des postes de travail de la ligne sur lesquels le produit n'est pas encore passé et qui peuvent donc voir leur allocation modifiée. (Le poste impacté contiendra alors des tâches non déplaçables qui occuperont de la place, mais il pourra accueillir d'autres tâches)

IV.3.3. Objectif

L'allocation est donnée par la fonction surjective $A_w: T \rightarrow W$. Pour cela, elle associe à chaque tâche $t(i)$ de T , le poste de travail $w(k)$ sur lequel celle-ci est allouée : $A_w(t(i)) = w(k)$. L'équilibrage initial correspond à une définition de cette fonction. La réallocation des tâches consiste alors à trouver une nouvelle définition de cette même fonction.

Une allocation d'une tâche sur un poste de travail est possible uniquement si la charge de travail restant sur un poste le permet, c'est-à-dire si la durée du travail sur ce poste n'excède pas le takt time (l'objectif est donc que la somme des durées des tâches soit le plus près possible du takt time, et cela pour chacun des postes de travail de la ligne). Notons que l'écart de cette somme au takt time est nommé ϵ . Dans ce manuscrit, nous privilégions la rapidité de résolution à l'optimalité, c'est pourquoi ϵ n'atteindra pas forcément sa valeur minimale.

La reconfiguration (le rééquilibrage de la ligne d'assemblage) consiste à trouver une solution acceptable, c'est-à-dire une nouvelle allocation A_w qui respecte les contraintes présentées ci-dessous.

Lors de la recherche d'un équilibrage optimal, le but est de minimiser ϵ . Dans nos travaux, notre objectif est de trouver une solution possible qui permet de poursuivre la production sans perte de capacité.

- La durée d'un poste $d_{w(i)}$ est calculé par la somme des durées des tâches allouées à ce poste, c'est à dire par la formule suivante $\forall w(k) \in W, d_{w(k)} = \sum_{t(i) \in T} (d_{t(i)} x_{ij})$ avec x_{ij} , une variable booléenne indiquant si la tâche $t(i)$ est allouée au poste $w(j)$:

$$\forall t(i) \in T, \forall w(k) \in W, x_{ij} = \begin{cases} 1 \text{ si } A_w(t(i)) = w(j) \\ 0 \text{ sinon} \end{cases}$$
- Le but est que cette durée soit inférieure au takt time, pour tous les postes de la ligne
 $\forall w(k) \in W, d_{w(k)} \leq \text{Takt Time}$

IV.3.4. Contraintes d'allocation

- Une tâche doit être allouée à un et seulement un poste de travail : $\forall t(i) \in T, \sum_{w(j) \in W} x_{ij} = 1$
- Seules les tâches de T_{bis} peuvent être déplacées. Donc pour les autres tâches (celles de $T - T_{bis}$), $A_w(t(i))$ est constant $\forall t(i) \in (T - T_{bis}), A_w(t(i))$ est constant
- Seuls les postes de W_{bis} peuvent accueillir de nouvelles tâches. Ainsi $A(t_i)$ est forcément différent de W_j $\forall t(i) \in T, A_w(t(i)) \neq w(j)$

IV.3.5. Contraintes de précédence

Pr est la matrice de précédence. Pour un couple de tâches $(t(i), t(j)) \in T^2$, Pr_{ij} vaut "1" si $t(i)_i$ doit précéder $t(j)$.

$$\forall t(i) \in T, t(j) \in T, Pr_{ij} = \begin{cases} 1 \text{ si } t(i) \text{ précède } t(j) \\ 0 \text{ sinon} \end{cases}$$

IV.3.6. Contraintes de parallélisation

Les contraintes de parallélisation précisent si deux tâches peuvent être réalisées en parallèle. En effet, les tailles des postes de travail et des produits à assembler étant assez importantes, deux opérateurs peuvent travailler simultanément et sans se gêner sur un même poste de travail. Soit P_a cette matrice de parallélisation :

$$\forall t(i) \in T, t(j) \in T, P_{a_{ij}} = \begin{cases} 1 & \text{si } t(i) \text{ et } t(j) \text{ peuvent être réalisées en parallèle} \\ 0 & \text{sinon} \end{cases}$$

(Les ressources ne sont pas traitées ici : coûts, changer la durée serait aussi une forme de coût : pour simuler la gêne occasionnée)

V. Conclusion

Après avoir présenté le concept d'équilibrage d'une ligne d'assemblage, un problème plus spécifique a été identifié : le rééquilibrage dynamique. En effet, pour faire face aux aléas le système doit être réagir rapidement. Le rôle du rééquilibrage est donc, lorsqu'un aléa est détecté, de trouver une nouvelle allocation des tâches afin d'assurer la production. En parallèle, cet événement doit être enregistré afin d'éventuellement demander un recalcul centralisé d'un équilibrage complet.

Les hypothèses et les travaux précédents concernant le système d'information permettant de traiter la détection du retard et l'application de la solution, le travail de cette thèse se concentre sur le problème de la recherche d'une solution de rééquilibrage. La formalisation de ce problème a permis d'identifier que, bien que peu étudié dans la littérature, il s'inscrit dans la classe des problèmes d'équilibrage généraux (GALBP, General Assembly Line Balancing).

Le problème étant identifié et formalisé, le chapitre suivant s'intéresse aux différentes méthodes pour le résoudre. Les méthodes de la littérature traitant de problèmes similaires seront donc étudiées, et deux de ces méthodes seront testées plus en détails. Cette évaluation nous permettra de conclure le chapitre 2 avant de passer à la proposition principale de la thèse.

Chapitre 2 : Les méthodes de résolution des ALBP

Table des matières

I. Introduction.....	37
II. État de l'art sur les méthodes de résolution ALBP.....	38
II.1. Méthodes exactes.....	38
<i>II.1.1. Programmation dynamique.....</i>	<i>38</i>
<i>II.1.2. Procédure par séparation et évaluation.....</i>	<i>38</i>
II.2. Méthodes approchées.....	39
<i>II.2.1. Méthodes constructives.....</i>	<i>40</i>
<i>II.2.2. Méthodes incrémentales.....</i>	<i>40</i>
II.3. Rééquilibrage des lignes d'assemblage.....	41
II.4. Conclusion de l'état de l'art.....	43
III. « Iterated Local Search » (ILS).....	43
III.1. Présentation du principe.....	43
III.2. Les différents types de voisinage et les critères d'arrêt.....	44
<i>III.2.1. Types de voisinage.....</i>	<i>44</i>
<i>III.2.2. Critères d'arrêt.....</i>	<i>44</i>
III.3. Formalisation du problème.....	45
<i>III.3.1. Variables et paramètres du modèle.....</i>	<i>45</i>
<i>III.3.2. Contraintes.....</i>	<i>45</i>
<i>III.3.3. Objectif.....</i>	<i>45</i>
<i>III.3.4. Les différents types de voisinages.....</i>	<i>46</i>
<i>III.3.5. Présentation des perturbations.....</i>	<i>46</i>
III.4. L'algorithme.....	46
IV. Recherche d'atteignabilité sur un réseau d'automates.....	48
IV.1. Introduction.....	48
IV.2. Historique des méthodes utilisant la théorie des SED.....	48
IV.3. Présentation du formalisme utilisé.....	49
<i>IV.3.1. Les automates.....</i>	<i>49</i>
<i>IV.3.2. Un réseau d'automates.....</i>	<i>50</i>

IV.4. Modélisation du problème de rééquilibrage.....	52
<i>IV.4.1. Les tâches.....</i>	<i>52</i>
<i>IV.4.2. Les postes de travail.....</i>	<i>52</i>
<i>IV.4.3. La ligne d'assemblage.....</i>	<i>53</i>
IV.5. La recherche d'atteignabilité.....	53
<i>IV.5.1. Principe.....</i>	<i>53</i>
<i>IV.5.2. Définition de la propriété.....</i>	<i>55</i>
V. Comparaison des 2 méthodes.....	55
V.1. Présentation du contexte et des instances.....	55
V.2. Évaluation des instances.....	58
<i>V.2.1. Les indicateurs.....</i>	<i>58</i>
<i>V.2.2. Valeur des indicateurs.....</i>	<i>58</i>
V.3. Évaluation des résultats obtenus.....	61
<i>V.3.1. Les éléments d'évaluation.....</i>	<i>61</i>
<i>V.3.2. Scénario 1 :.....</i>	<i>62</i>
<i>V.3.3. Scénario 2 :.....</i>	<i>63</i>
<i>V.3.4. Scénario 3 :.....</i>	<i>64</i>
<i>V.3.5. Scénario 4 :.....</i>	<i>66</i>
V.4. Analyse des résultats.....	66
VI. Conclusion.....	67

I. Introduction

Les premiers modèles analytiques pour résoudre le problème de l'équilibrage sont apparus dans les années 50 (Salveson, 1955 ; Helgeson and Birnie, 1961). Depuis les méthodes de résolution ont été améliorées et les modèles complétés pour prendre en compte de nouveaux éléments par exemple : l'ajout de nouvelles contraintes, des adaptations des objectifs, la prise en compte des durées de tâches aléatoires, etc.

De nombreux états de l'art sur les méthodes de résolution, quelles soit exactes ou approchées, ont été présentés dans (Baybars, 1986 ; Ghosh and Gagnon, 1989 ; Erel and Sarin, 1998 ; Rekiek et al., 2002 ; Becker and Scholl, 2006 ; Dolgui and Proth, 2006 ; Scholl and Becker, 2006 ; Boysen et al., 2008).

Comme nous l'avons introduit, malgré la richesse de la littérature, les auteurs tentent toujours de trouver de nouvelles méthodes de résolution de ce problème, car il est toujours possible d'améliorer la vitesse de résolution, d'augmenter l'optimalité de la solution trouvée, ou de prendre en compte de nouvelles contraintes (durées des tâches variables de manière aléatoire (McMullen and Frazier, 1997), de nouveaux objectifs (prise en compte de l'espace en bord de ligne pour l'allocation des tâches (McMullen and Frazier, 1997 ; Bautista and Pereira, 2007), etc.).

Ce problème est à rapprocher de problèmes d'allocations tels que les problèmes de sac-à-dos ou d'ordonnancement qui sont des problèmes NP-Complets. Cette complexité limite les performances des méthodes exactes. En effet, pour assurer l'optimalité de la solution fournie, ces méthodes ont besoin d'un temps de calcul augmentant avec le nombre de tâches à réallouer. Leur application à de grandes instances devient donc vite problématique. (Falkenauer, 2005) ont montré ainsi que ces méthodes exactes sont difficilement applicables aux cas industriels.

Pour s'affranchir de ce problème d'explosion combinatoire, des méthodes d'approximation ont été développées, nommées heuristiques ou méta heuristiques. Celles-ci procèdent par étape, dans le but de fournir une solution la plus proche possible de la solution optimale en construisant une solution puis en l'évaluant. Il s'agit donc de trouver un compromis entre l'optimalité de la solution et le temps accordé pour la résolution. En effet, il est plus rapide de vérifier qu'une allocation donnée respecte les contraintes, que de trouver l'allocation optimale.

Le but de ce chapitre est donc d'étudier les différentes méthodes de résolution proposées dans la littérature pour résoudre le problème du rééquilibrage dynamique d'une ligne d'assemblage de type GALBP (reconfiguration avec la possibilité de paralléliser les tâches) ou des problèmes équivalents.

Nous étudierons plus particulièrement deux méthodes approchées : la méthode ILS (Iterated Local Search) et la recherche d'atteignabilité sur un réseau d'automates communicants. La conclusion de cette étude nous montrera que les deux approches peuvent répondre au besoin de réactivité imposé par notre problème avec un avantage à la recherche d'atteignabilité sur un réseau d'automates communicants, notamment lorsque l'on souhaite privilégier différents leviers d'actions.

II. État de l'art sur les méthodes de résolution ALBP

II.1. Méthodes exactes

Pour résoudre les problèmes d'ALBP, le domaine de la recherche opérationnelle propose plusieurs méthodes exactes. La plupart des méthodes de résolution utilisent des modèles de programmation linéaire (en nombre entier ou mixte s'il y a des valeurs réelles). Les méthodes de résolution proposées se divisent en deux grandes familles : la programmation dynamique et les algorithmes de séparation et d'évaluation aussi connus sous leurs nom anglais « branch and bound ».

II.1.1. Programmation dynamique

La programmation dynamique (Bellman, 1952) est une méthode classique de résolution utilisée dans de nombreux problèmes d'optimisation (Dreyfus and Law, 1977) aussi variés que les problèmes d'ordonnancement (Held and Karp, 1962) ou pour la reconnaissance de mots (Sakoe and Chiba, 1978).

Le principe de cette méthode est d'explorer l'espace des solutions de manière intelligente en divisant le problème en sous-problèmes. Cela consiste à résoudre les sous-problèmes et en combinant leurs solutions nous obtenons la solution globale selon le principe d'optimalité de Bellman. Cette méthode s'applique à des problèmes d'optimisation dont la fonction objectif est additive. Lors de l'exploration, les sous problèmes sont alors analysés uniquement par le résultat de leur résolution, sans besoin de rentrer dans le détail de leur construction. Par exemple, pour le calcul d'un plus court chemin dans un graphe, on peut utiliser chaque nœud comme points intermédiaires en ne retenant à chaque fois que le plus court chemin entre ce nœud et le nœud initial, et ne pas retenir les autres chemins que nous savons plus longs.

Cette méthode a été appliquée au problème SALBP-1 pour la première fois par (Jackson, 1956) puis par (Held et al., 1963). Les auteurs proposent un modèle de résolution sous forme de graphe orienté station (équivalent des postes de travail des lignes automatisées). Chaque nœud représente les différentes allocations possibles. L'allocation des tâches est alors effectuée par les arcs dont le poids correspond à la durée de chaque tâche. La résolution passe alors par la recherche du plus court chemin.

D'autres modèles orientés tâches ont été proposés. Dans ce cas, chaque noeud correspond à l'allocation d'une tâche. La solution est aussi trouvée par une recherche du plus court chemin (Schrage and Baker, 1978).

Ces modèles ont été aussi utilisés dans des travaux plus récents (Gutjahr and Nemhauser, 1964 ; Schrage and Baker, 1978 ; Bard, 1989 ; Easton et al., 1989 ; Erel and Gokcen, 1999 ; Dolgui et al., 2006 ; Dolgui et al., 2008)].

II.1.2. Procédure par séparation et évaluation

Le principe de la procédure par séparation et évaluation (PSE), connue sous l'appellation anglo-saxonne de Branch and Bound est de limiter l'exploration de l'espace de solutions afin de ne parcourir que les chemins pouvant mener à la solution optimale. Pour cela, elle se base sur 2 étapes : séparation et évaluation.

Tout d'abord l'espace de solutions du problème est divisé en sous-ensembles (formant un recouvrement pour être certains d'analyser toutes les solutions possibles). Chacun de ces sous-ensembles

est alors évalué et ceux dont on est certain qu'ils ne contiennent pas de solutions ne sont pas explorés et les autres réévalués au niveau suivant. L'évaluation est donc une étape clé de cette méthode : un mauvais choix conduisant à plus de recherches inutiles rallongera la recherche globale, à l'inverse la suppression de certaines branches trop tôt empêchera d'atteindre la solution optimale. Dans les cas extrêmes, si la sélection est trop lâche, il se peut que tout l'arbre des possibles soit exploré causant alors une explosion combinatoire incompatible avec les grandes instances. (Land and Doig, 1960). (Lawler and Wood, 1966) proposent un état de l'art des différentes variantes de PSE pouvant être utilisé à la place de la programmation dynamique.

Pour le domaine de l'équilibrage de ligne d'assemblage, dans la littérature, quatre méthodes importantes mettant en œuvre la PSE ont d'abord été proposées pour résoudre les problèmes de SALBP :

- La procédure FABLE (Fast Algorithm for Balancing Lines Effectively) a été proposée par (Johnson, 1988). Cette procédure utilise une sélection orientée tâche, c'est à dire que chaque séparation est réalisée par l'allocation d'une seule tâche sur le poste de travail considéré. Les sous-problèmes obtenus diffèrent donc de la solution de niveau supérieur par la modification de l'allocation d'une seule tâche.
- La méthode EUREKA a été proposée par (Hoffmann, 1992). Celle-ci est orientée poste de travail. À chaque séparation, les sous-problèmes sont obtenus en sélectionnant un ensemble de tâches qui sont affectées sur un poste de travail.
- L'algorithme SALOME (Simple Assembly Line Balancing Optimization Method) a été proposé par (Scholl and Klein, 1997). Cet algorithme combine les approches utilisées dans FABLE et EUREKA. Il a été appliqué au SALBP-2 par (Klein and Scholl, 1996).
- La méthode OptPack (Nourie and Venta, 1991) est une autre PSE orientée tâche comme FABLE. Cependant, OptPack utilise l'heuristique de (Hoffmann, 1992) pour améliorer le parcours de l'arbre des solutions.

Les méthodes citées ci-dessus sont les plus connues et ont été comparées par (Scholl and Klein, 1999). Cependant, il en existe de nombreuses autres. Ainsi (Scholl et al., 2010) proposent ABSALOM qui est une méthode inspirée de SALOME permettant de résoudre des SALBP de plus grandes dimensions mais aussi de prendre en compte les contraintes de restriction des assignations. Les PSE sont aussi utilisées pour la prise en compte de contraintes supplémentaires à l'ALBP telles que le mixed-model de la ligne d'assemblage (Bukchin and Rabinowitch, 2006) ou l'allocation des opérateurs (Miralles et al., 2008).

De manière plus spécifique, on trouve aussi une PSE pour résoudre les problèmes de SALBP-2. (Hackman et al., 1989) proposent une méthode de recherche qui résout le SALBP-2 en s'appuyant sur une succession de résolutions d'instances de SALBP-1. Cette itération est inspirée de (Helgeson and Birnie, 1961).

II.2. Méthodes approchées

Il existe aussi un nombre important de travaux proposant des heuristiques pour résoudre le SALBP. Des études comparatives des performances des différentes heuristiques pour l'équilibrage des lignes

d'assemblage ont été effectuées dans (Talbot et al., 1986 ; Amen, 2000 ; Capacho et al., 2007). Nous pouvons les organiser en deux grandes familles : les méthodes constructives et les méthodes incrémentales.

II.2.1. Méthodes constructives

Une méthode constructive permet de trouver la solution d'un problème d'optimisation en progressant par étapes successives. Par exemple, pour résoudre un ordonnancement, la méthode commencera par placer un premier produit, puis un second, tout en s'assurant que les contraintes sont bien respectées et que la solution vers laquelle on se dirige atteindra l'objectif. Après plusieurs itérations, tous les éléments auront alors été pris en compte, et la solution sera trouvée.

Les méthodes approchées les plus utilisées pour résoudre les problèmes d'équilibrage sont COMSOAL (COMputer Method for Sequencing Operations for Assembly Lines) proposées par (Arcus, 1965) et RPW (Ranked Positional Weight) (Helgeson and Birnie, 1961).

(Arcus, 1965) propose une version stochastique d'une heuristique multi-passages, la procédure génère plusieurs solutions complètes jusqu'à ce qu'un certain nombre soit atteint. À chaque itération, une opération à affecter est choisie aléatoirement à partir de la liste des tâches disponibles, c'est-à-dire celles qui ont leurs prédécesseurs déjà affectés et dont le temps d'exécution est inférieur au temps restant sur le poste de travail courant. Le processus est réitéré jusqu'à l'obtention d'une solution complète. À l'évidence, plus le nombre de solutions construites est important, plus élevées sont les chances de trouver l'optimum (ou une solution proche).

Comme son nom l'indique, l'heuristique RPW (Ranked Positional Weight) se base sur l'affectation de poids aux différentes tâches, celles-ci étant ensuite classées selon ce critère. Ce poids dépend de la durée nécessaire pour effectuer la tâche proprement dite, mais aussi de celle qui la précède et de celle qui lui succède. Elles sont ensuite classées dans l'ordre décroissant de ce paramètre. Cet ordre est utilisé pour l'affectation de celles-ci sur les postes de travail. Les tâches qui ont le plus grand poids seront donc affectées de manière prioritaire sur les premiers postes de la ligne.

Il existe aussi les algorithmes d'optimisation par colonies de fourmis comme proposé par (Bautista and Pereira, 2007). Le principe ici est de construire plusieurs solutions. La solution optimale est alors sélectionnée par l'augmentation de son poids par plusieurs sélections. Les autres solutions finissent par disparaître. Cela est inspiré du comportement des fourmis où le plus court chemin apparaît par l'accumulation de phéromones et par la sélection de la piste qui en a le plus.

II.2.2. Méthodes incrémentales

Ces méthodes sont basées sur l'amélioration successive des solutions qui peuvent initialement être obtenues avec des méthodes constructives. Par exemple, les algorithmes génétiques manipulent une population de solutions grâce à plusieurs opérateurs de type croisement, sélection et mutation. Leur application fait apparaître quelques difficultés telles que le codage des solutions, le lissage de la fonction fitness et la réparation des solutions rendues irréalisables après l'application de croisements (Kim et al., 1996) (Pierreval et al., 2003) (Falkenauer, 1992).

Un autre algorithme bio-inspiré a été proposé pour résoudre le problème SALBP : l'optimisation par essais particuliers. La solution est construite à partir de l'interaction de proche en proche entre les

différentes particules. En effet, chaque élément de l'essaim adapte sa position en fonction de son expérience, mais aussi en fonction de l'expérience de ses voisins directs qui font de même. Ainsi, chaque élément modifie sa vitesse et sa direction en fonction des particules proches. Cette information se diffuse à travers l'essaim avec une certaine vitesse. Finalement, l'ensemble est modifié. Il s'agit donc d'une méthode incrémentale qui permet de parcourir l'espace de solutions (Kalayci and Gupta, 2013). Un autre exemple est aussi donné par (Chutima and Chimklai, 2012).

Pour résoudre le SALB, une autre meta-heuristique a été proposée par (Chiang, 1998) en utilisant la méthode de recherche tabou (Glover, 1989 ; Glover, 1990). Cette méthode a ensuite inspiré (Lapierre and Ruiz, 2004 ; Lapierre et al., 2006) qui ont proposé une méthode pour résoudre des cas industriels. Leur méthode utilise de manière complémentaire deux techniques de recherche locale pour l'intensification et la diversification. De plus, ils proposent des modifications de l'objectif et du parcours de l'espace de recherche afin d'explorer des solutions non valides, c'est à dire qui ne respectent pas les contraintes de temps de cycle.

Les méthodes tabou ont ensuite été utilisées pour des problèmes de GALBP comme par exemple (Özcan and Toklu, 2008) qui proposent de résoudre le problème en prenant en compte des lignes où les opérateurs travaillent par paires sur un même produit en étant placés de chaque côté de la ligne (« Two-sided assembly lines »).

II.3. Rééquilibrage des lignes d'assemblage

En 2005, (Falkenauer, 2005) a mis en évidence une des lacunes de la littérature qui traite plus le problème de la conception d'une ligne d'assemblage que de la reconfiguration de celle-ci. Or dans la vie d'une usine, les entreprises sont plus confrontées à la reconfiguration. Cette reconfiguration a pour but d'adapter les lignes de production existantes pour l'arrivée de nouveaux produits ou de changement de processus. En effet, dans la pratique, la ligne de production des nouveaux produits sera basée sur les lignes d'assemblage déjà existantes : nous sommes alors face à un problème de reconfiguration plutôt que de conception.

Lors de la conception d'une ligne d'assemblage, son équilibrage est réalisé. Dans le cas d'une reconfiguration, on réalise de même cet équilibrage, en prenant en compte divers contraintes et objectifs de réutilisation des éléments existants. On parle alors de rééquilibrage de ligne d'assemblage dont l'un des premiers modèles a été présenté en 2006 par (Gamberini et al., 2006). Les auteurs introduisent notamment le facteur de similitude permettant de mesurer l'écart entre l'ancienne ligne d'assemblage et la ligne avec son nouveau rééquilibrage. Le but étant alors de réduire les coûts de reconfiguration en restant proche de la ligne initiale, tout en minimisant les coûts de production en améliorant l'équilibrage et en tenant compte des nouvelles données (par exemple la durée des tâches peut évoluer avec l'effet d'apprentissage des opérateurs).

Pour résoudre ce problème d'optimisation multicritères, ils proposèrent de coupler une heuristique proposée par (Kottas and Lau, 1973) avec la méthode « Technique for Order Preference by Similarity to Ideal Solution » (TOPSIS). En 2009, ils ont ensuite proposé de résoudre ce même problème avec un algorithme génétique (Gamberini et al., 2009).

(Altemeier et al., 2010) se placent dans le contexte des RMS (Reconfigurable Manufacturing Systems) en considérant une ligne d'assemblage d'automobiles assemblant une grande variété de produits. Leur but est de trouver une reconfiguration de la ligne existante afin d'en améliorer son efficacité tout en minimisant le coût de reconfiguration.

(Makssoud et al., 2013 ; Makssoud et al., 2014a ; Makssoud, 2014b) proposent l'optimisation combinatoire pour reconfigurer une ligne de transfert. Leur but est alors de concevoir une ligne de transfert équilibrée en utilisant au mieux les machines déjà existantes. Pour cela ils fixent des coûts de déplacement, de modification, etc.

D'autres auteurs traitent le problème de rééquilibrage en prenant en compte l'utilisation d'opérateurs supplémentaires. On retrouve cela dans (Corominas et al., 2008). Ces opérateurs posent des contraintes supplémentaires car ils doivent être accompagnés d'opérateurs inexpérimentés dont le temps mis pour réaliser une tâche est plus élevé (donc leurs coûts aussi). Il faut donc minimiser le nombre d'opérateurs tout en assurant l'équilibrage. La prise en compte d'opérateurs temporaires est encore étudié comme par exemple dans (Kim et al., 2015).

Comme nous l'avons dit précédemment, les reconfigurations ont lieu afin d'inclure de nouveaux produits ou des améliorations majeures des processus de lignes de production. Celles-ci ont donc lieu de manière occasionnelle et sont prévues pour être valables pendant plusieurs saisons de production. Leur horizon temporel est donc supérieur au mois. Nous pouvons donc situer la reconception au sommet du schéma ci-dessous inspiré par (Scholl, 1999) et (Grangeon et al., 2011) (Figure 6).

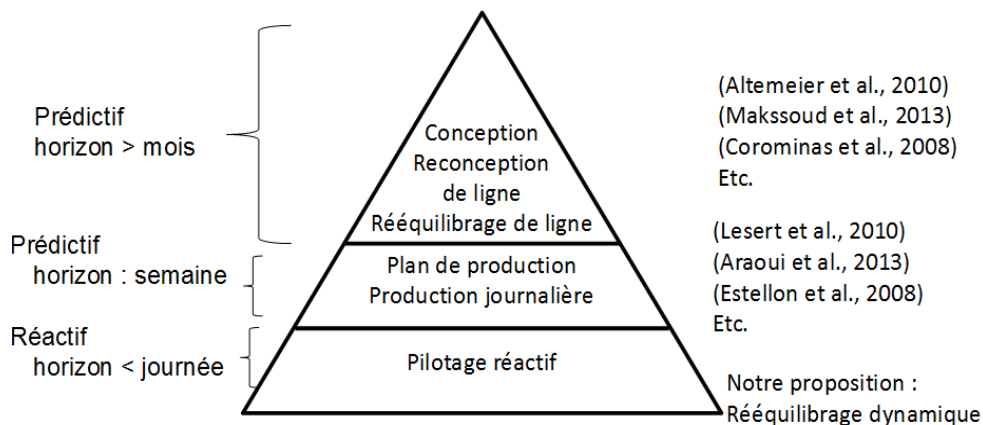


Figure 6: Les différents types de reconfiguration de ligne et leurs horizons temporels.

Pour un horizon temporel inférieur au mois, reconfigurer la ligne consiste dans la plupart des cas à modifier l'ordonnancement de la ligne. C'est ce qui est réalisé dans l'automobile, où les postes glissants et une séquence de produits à assembler correctement définie, permettent d'absorber les pics de charges et ainsi de rééquilibrer la ligne. L'équilibrage est donc défini, et c'est le séquençement qui doit permettre le lisage correct de la ligne en assurant le respect des contraintes d'espacement entre les véhicules. C'est le cas dans les travaux de (Lesert et al., 2010 ; Lesert et al., 2011 ; Araoui et al., 2013) et le thème du challenge roadef 2005 <http://challenge.roadef.org/2005/fr/> dont les gagnants ont publié leurs résultats dans (Estellon et al., 2008). Ce film de production est réalisé pour la semaine, puis adapté au jour le jour.

A ce jour nous n'avons pas trouvé des travaux traitant de la reconfiguration dynamique dans un horizon court voir très court terme. Notre proposition vient donc combler ce manque.

II.4. Conclusion de l'état de l'art

Nous avons présenté dans cette première partie du chapitre, un panorama des méthodes exactes et approchées utilisées pour la résolution des problèmes de rééquilibrage de ligne d'assemblage. Nous avons également pu mettre en évidence le manque d'études abordant le problème du rééquilibrage dynamique. Les deux types de méthodes de résolution (exactes ou approchées) présentent des inconvénients et des avantages qui leur sont propres. Le choix s'effectuera en fonction des objectifs visés (généralement la qualité de la solution ou le temps de résolution). Concernant les méthodes exactes le principal problème est leur temps de résolution qui est souvent long. Leur application est contrainte ou limitée par la taille et la nature du problème. Par contre elles ont l'avantage de garantir de l'optimalité. Les méthodes approchées ont l'avantage de pouvoir proposer une solution proche de l'optimum dans un temps de résolution court (quelques secondes). Leur principal point faible réside dans le manque de robustesse de la qualité des solutions apportées par ces algorithmes. Même lorsqu'une méthode approchée est efficace pour un type de problème, il reste difficile de garantir que la qualité des résultats obtenus soit la même pour des données différentes.

Dans cette thèse, nous avons fait le choix d'une résolution approchée car elle permet de répondre à la contrainte de réactivité imposée par l'horizon de décision court terme d'un rééquilibrage dynamique. La taille des instances et le besoin de traiter plusieurs situations industrielles différentes nous confortent également dans ce choix. Mais par contre nous souhaitons proposer une solution qui sera capable d'assurer un certain niveau de qualité de la solution lors de répétitions de mise en œuvre.

Dans la suite du chapitre nous allons comparer deux méthodes approchées prometteuses qui n'ont pas été étudiées dans le domaine ALBP: Il s'agit d'une méthode issue du domaine de la recherche opérationnelle (ILS) et une autre méthode issue du domaine du contrôle, la recherche d'atteignabilité sur un réseau d'automates communicants. Nous avons fait le choix d'étudier plus en avant ces méthodes, car elles permettent de pallier les faiblesses de celles présentées précédemment.

III. « Iterated Local Search » (ILS)

III.1. Présentation du principe

La méthode « ILS » est une méta-heuristique fournissant un cadre simple mais efficace pour améliorer les performances des méthodes de recherche locale (voisinage). Sa simplicité, son efficacité et son efficience ont intéressé plusieurs chercheurs de domaines variés. Ainsi elle a été utilisée pour résoudre le problème de voyageur de commerce (Stützle and Hoos, 2002), le routage d'une flotte de véhicules (Penna et al., 2011) ou encore le séquençement d'atelier (Stützle, 1998). À ce jour, nous n'avons pas trouvé de travaux appliquant cette méthode dans le domaine du rééquilibrage.

La méthode ILS est une méthode de voisinage qui conduit aussi à introduire des perturbations permettant de sortir des optimums locaux en testant une autre partie de l'espace de solutions.

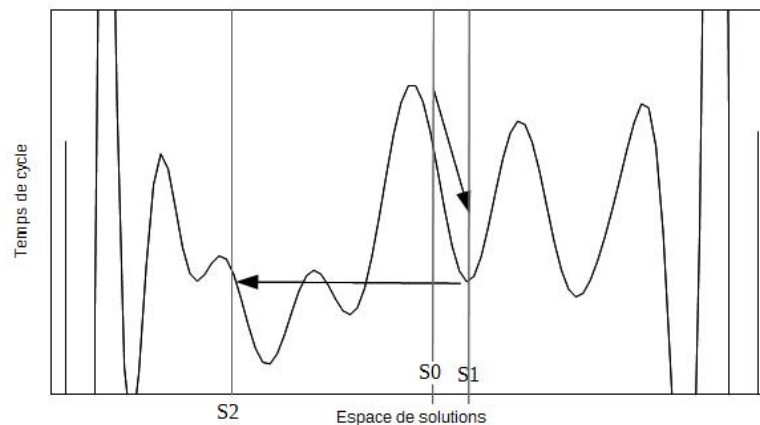


Figure 7: Principe de fonctionnement de ILS.

La figure ci-dessus (Figure 7) présente le principe général de la méthode ILS. La première étape consiste à appliquer une procédure de recherche locale à partir d'une solution S_0 permettant de trouver une solution S_1 plus proche de l'optimal. On applique ensuite une perturbation sur S_1 , nous permettant d'obtenir une nouvelle solution S_2 sur laquelle on applique de nouveau la procédure de recherche locale, puis une perturbation. Ces différentes itérations sont répétées jusqu'au critère d'arrêt permettant ainsi de parcourir une plus grande étendue de l'espace de solutions afin de s'assurer que l'optimum trouvé n'est pas un optimum local et donc est le plus proche de la solution optimale.

Nous allons voir maintenant comment les méta-heuristiques utilisant des méthodes par voisinage (comme la recherche Tabou ou la méthode Kangourou), définissent les voisinages et les critères d'arrêt dans le cas du problème SALBP-2. Puis nous proposerons une méthode de résolution de notre problème avec ILS.

III.2. Les différents types de voisinage et les critères d'arrêt

III.2.1. Types de voisinage

D'après (Moodie, 1964 ; Rachamadugu and Talbot, 1991), les méta-heuristiques de type recherches locales utilisées pour résoudre les SALBP de type 2 proposent deux types de voisinages :

- Déplacement d'une tâche t_i d'un poste w_1 à un poste w_2 .
- Permutation de deux tâches de deux postes différents.

Nous verrons en quoi ces types de voisinages sont en accord avec la méthode ILS que nous développerons pour notre problème.

III.2.2. Critères d'arrêt

(Heinrici, 1994) et (Scholl and Voß, 1997) ont créé une procédure de recherche tabou pour la résolution de problèmes de type SALB. Ils ont intégré au problème des critères d'arrêt de la procédure :

- Après un certain nombre d'itérations,
- Au bout d'un certain temps, (compromis durée résolution / optimalité)
- Quand le temps de cycle maximal est égal au temps de cycle optimal calculé.

En s'inspirant de ces travaux, notre approche calculera ce temps de cycle optimal en prenant en compte la perturbation engendrée par les aléas. Les 2 autres critères d'arrêts seront fixés avant le lancement de la résolution.

III.3. Formalisation du problème

III.3.1. Variables et paramètres du modèle

- n nombre des tâches à allouer.
- p nombre de postes.
- TT takt time objectif.
- i indice sur les tâches ($\in [1, n]$).
- j indice sur les postes ($\in [1, p]$).
- t_i temps opératoire de la tâche i .
- x_{ij} : 1 si la tâche i est affectée au poste j , 0 sinon (c'est la formulation de notre solution puisqu'elle représente l'allocation des tâches).
- Le retard d'un poste j sera vécu comme l'augmentation d'un Δt_j .
- $\sum_{i=1}^n (x_{i,j} t_i) + \Delta t_j$, c'est la durée pendant laquelle chaque poste j est occupé.
- P matrice de précédence. Pour chaque paire de tâches (i, j) on a $P_{i,j}$ égal à 1 si la tâche i est un prédécesseur de la tâche j , 0 sinon.

III.3.2. Contraintes

- Chaque tâche doit être affectée à un poste : $\forall i, \sum_{j=1}^p x_{i,j} = 1$
- Chaque tâche doit respecter la contrainte de la précédence. Autrement dit, une tâche i ne peut être affectée à un poste j que si tous ses prédécesseurs ont été déjà affectés à des postes en amont du poste j .

III.3.3. Objectif

L'objectif est d'atteindre un temps de cycle inférieur ou égal au TT en intégrant le temps du retard Δt_j .

$$\text{Soit } \forall j, \sum_{i=1}^n (x_{i,j} t_i) + \Delta t_j \leq TT$$

De plus, le coût de mise en place du nouvel équilibrage (nombre de changements) pour chaque sous-résultat trouvé est enregistré. Ainsi, on arrête si la solution devient plus importante que la perte due au retard.

III.3.4. Les différents types de voisinages

Pour mettre en place la procédure de recherche locale nous proposons trois types de voisinages présentés ci-dessous. Le plus petit voisinage que nous pouvons imaginer, celui perturbant le moins la ligne est celui où un seul poste est impacté. Cependant, cela ne permet pas de modifier l'équilibrage dans le cas où les tâches sont effectuées de façons séquentielles. Il faut donc considérer deux postes.

a. Permutation de 2 tâches (une tâche échangée avec une autre)

Ce voisinage consiste à choisir aléatoirement une tâche du poste bloquant (le poste qui a le plus grand temps de cycle) et qui se retrouve après la tâche retardée et de la permuter avec une autre tâche qui a été aussi choisie aléatoirement parmi les tâches appartenant aux postes en aval de celui bloquant. (Ce voisinage pourrait être restreint au poste bloquant et à un seul poste aval, cependant, nous préférons minimiser le nombre de déplacement que la distance entre les deux postes impliqués dans la réallocation).

b. Déplacement aléatoire

Ce voisinage consiste à considérer aléatoirement, de façon similaire à celui qui précède, une tâche du poste bloquant et qui se retrouve après la tâche retardée et l'affecter à l'un des postes en aval de celui bloquant. On provoque ainsi une insertion d'une tâche sur un autre poste.

Ce voisinage a néanmoins une forte limitation. Il s'agit du fait qu'il nécessite l'existence d'un poste qui dispose d'une marge suffisante (différence entre le takt time et le temps de cycle du poste) supérieure à la durée de la tâche choisie. Cela empêche l'utilisation de ce voisinage de manière unique : il doit être combiné.

c. Permutations de 3 tâches (deux tâches échangées avec une autre)

Ce voisinage consiste à sélectionner aléatoirement 2 tâches différentes du poste bloquant qui se retrouvent après la tâche retardée et de les permuter avec une autre tâche choisie, elle aussi, aléatoirement parmi les tâches appartenant aux postes en aval de celui qui est bloquant.

D'ailleurs, ce voisinage, étant plus large que le précédent (déplacement de plusieurs tâches : on s'éloigne plus de l'équilibrage initial), va représenter la perturbation dans notre modèle final (ILS).

III.3.5. Présentation des perturbations

Comme expliqué précédemment, la méthode ILS est une méthode de voisinage couplée avec l'introduction de perturbations appliquées pour sortir des optimums locaux et de tester une autre partie de l'espace de solutions.

La perturbation doit être plus large que les voisinages choisis pour la procédure de recherche locale, c'est-à-dire déplacer plus de tâches : nous avons donc choisi d'utiliser le voisinage « permutations de 3 tâches » comme une perturbation.

III.4. L'algorithme

La figure suivante (Figure 8) présente les différentes étapes de notre méthode. Nous commençons par initialiser la tâche retardée ainsi que la durée du retard. Cela définira le poste bloquant et également la

solution initiale. On applique ensuite la recherche locale et les perturbations. Lorsqu'un critère d'arrêt est atteint, l'algorithme est stoppé et la dernière solution trouvée est retournée.

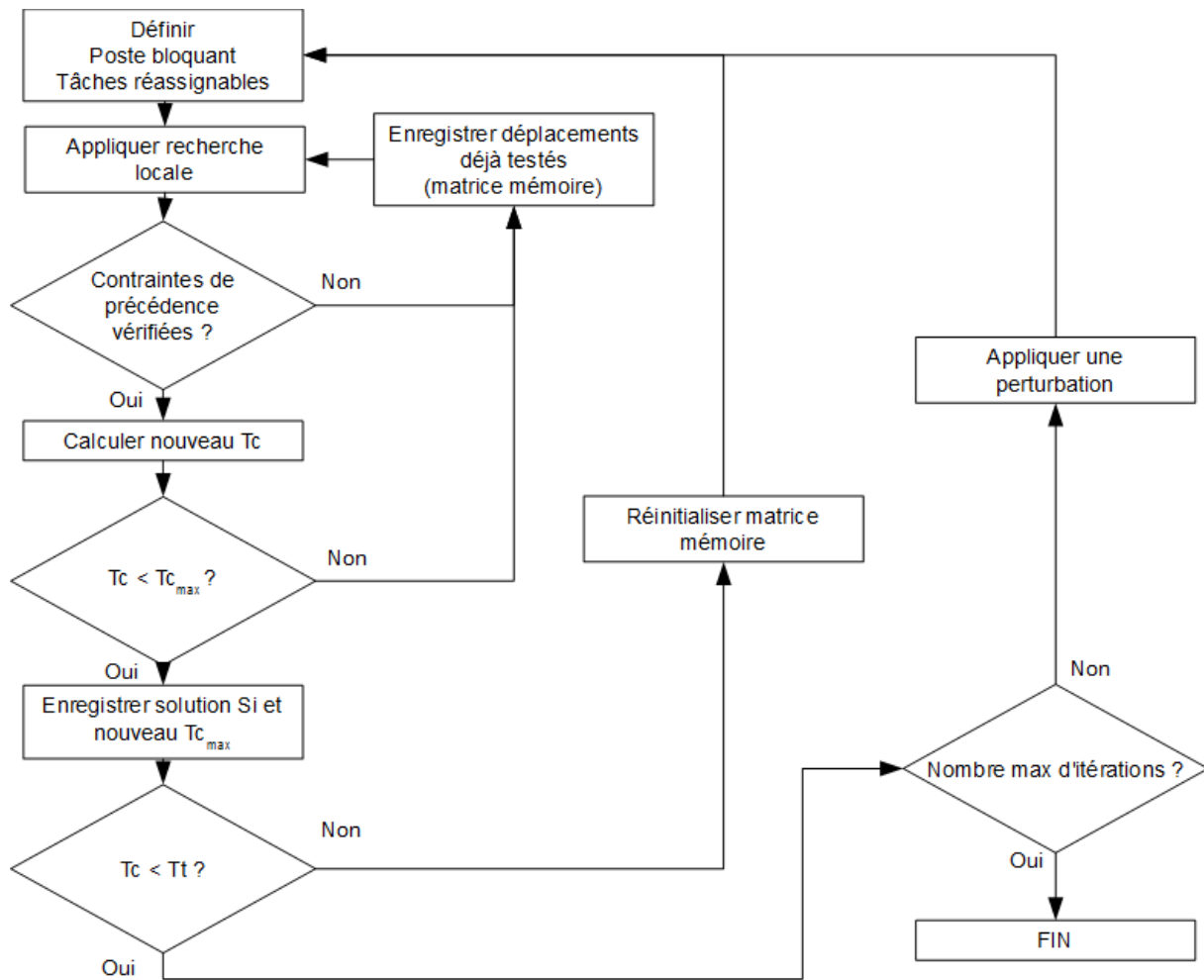


Figure 8: Algorithme de la méthode ILS.

L'algorithme permettant de réaliser la méthode ILS est présenté ci-dessous. Celui-ci inclut deux boucles principales et plusieurs tests permettant de trier les solutions et de surveiller les différents critères d'arrêt. Pour cela, nous définissons :

- k , le nombre d'itérations de la perturbation
- K_{max} , le nombre maximum d'itérations (critère d'arrêt)
- $Durée$, le temps écoulé depuis le début de l'algorithme
- $Durée_{max}$, le critère d'arrêt limitant le temps de résolution
- Tt , le takt time
- Tc , le temps de cycle de la solution évaluée
- Tc_0 , le temps de cycle de la solution précédente

- $T_{c_{max}}$, le temps de cycle maximum trouvé pour l'itération en cours (on cherche à le diminuer à chaque itération)
- La matrice solution représente la nouvelle allocation des tâches sur les postes

```
k := 0
Tant que k < Kmax
  Tant que (TC > TT) & (Durée < Duréemax)
    Choix des tâches à déplacer
    Si (contraintes de précédence == vrai) & (TC < TC0)
      Si (TC < TT)
        Solution trouvée
        Fin algorithme
      Sinon
        TC0 := TC
        Mise à jour matrice mémoire
    Sinon
      Écarter solution
      Mise à jour matrice mémoire
  Fin du tant que
  Choix des tâches à déplacer
  Permutation "2 tâches/ 1 tâche".
  Si TC < TC0
    TCmax := TC
    Enregistrement matrice solution
  k := k + 1
Fin du tant que
```

Cet algorithme a été implémenté en VBA avec une initialisation réalisée par des tableaux de valeurs saisis sur un tableur. Le code est fourni en annexe 1.

IV. Recherche d'atteignabilité sur un réseau d'automates

IV.1. Introduction

Dans cette partie, nous allons détailler une autre méthode déjà utilisée dans des problèmes similaires d'allocation : la recherche d'atteignabilité sur un réseau d'automates communicants. Cette méthode appliquées au problème de rééquilibrage de ligne d'assemblage a été présentés dans de précédents travaux (Manceaux et al., 2014a ; Manceaux et al., 2014b). Nous pourrons alors comparer les résultats obtenus par celle-ci par rapport à la méthode présentée précédemment.

Nous commençons par décrire l'origine de la méthode proposée en présentant l'historique des différentes modélisations et résolutions à l'aide de la théorie des Systèmes à Événements Discrets (SED). Cela nous permettra de justifier le choix de cette approche.

Ainsi par la suite nous décrivons le formalisme utilisé pour modéliser le problème. Ensuite, les détails du modèle sont expliqués. Enfin, la dernière partie explique comment une solution est trouvée en utilisant le modèle et la recherche d'atteignabilité.

IV.2. Historique des méthodes utilisant la théorie des SED

La théorie des Systèmes à événements discrets a été utilisée pour modéliser différents systèmes, soit en phase de conception, soit en phase d'usage pour son évaluation. Ces modèles peuvent alors être utilisés pour tester le comportement attendu du système considéré. Pour cela, deux utilisations de ces modèles coexistent : la simulation et la vérification formelle de propriétés.

Ainsi dans le domaine de l'équilibrage, (Vain et al., 2002) utilisent la simulation par des réseaux de Petri afin de vérifier le comportement d'un l'équilibrage calculé. En effet, après avoir calculé un équilibrage à l'aide d'une méthode de Programmation Linéaire en Nombres Entiers (PLNE), les auteurs souhaitent affiner la précision de leur solution. Pour cela, ils modélisent par un réseau de Petri le poste de la ligne qui est le plus chargé selon l'équilibrage calculé. Cette analyse plus détaillée du comportement du poste goulot (durée supplémentaire induite par les changements de mode opérationnels au sein du poste, etc.) permet de valider l'équilibrage précédemment calculé, ou de l'infirmer et de réclamer un nouveau calcul. L'approche SED est alors un complément de la PLNE.

Notre problème s'apparentant à un problème d'allocation, il est donc proche des problèmes d'ordonnancement. Pour ce type de problèmes, la vérification formelle de propriétés sur un modèle SED a été utilisée, et plus particulièrement la méthode que nous allons détailler : la recherche d'atteignabilité (Behrmann et al., 2005 ; Subbiah and Engell, 2010 ; Marangé et al., 2011).

Afin de valider leur approche, (Behrmann et al., 2005) proposent une comparaison de cette méthode par rapport à une méthode d'optimisation utilisant un modèle de Programmation Linéaire en Nombres Entiers Mixte (PLNE, MILP, Mixed Integer Linear Programming). Leurs conclusions annoncent que la recherche d'atteignabilité semble prometteuse pour résoudre ce type de problème : en effet, les performances obtenues sont similaires.

L'utilisation d'automates communicants mettant en œuvre une recherche d'atteignabilité pour les problèmes d'allocation a été élargie par (Lematre, 2013) pour traiter un problème d'allocation de fonctions sur un ensemble de contrôleurs.

Les problèmes d'équilibrage peuvent être assimilés à un problème d'allocations de tâches sur un ensemble de postes de travail, permettant ainsi de s'inspirer des méthodes décrites précédemment. Notre objectif de temps réel nous amène aussi à considérer cette piste comme prometteuse. Ainsi nous proposerons de tester l'utilisation des automates communicants formalisant les tâches et les opérations et donc le fonctionnement des réallocations. Les tâches sont allouées à une machine par le biais de la synchronisation entre les automates communicants. Ainsi, le modèle représente toutes les allocations possibles, et c'est lors de la recherche d'atteignabilité qu'une solution vérifiant les contraintes et les propriétés est trouvée.

Pour détailler cela, nous commencerons par décrire le formalisme utilisé pour la modélisation. Nous l'appliquerons ensuite à la modélisation de notre système. Puis nous verrons le principe de la recherche d'atteignabilité et comment celle-ci est utilisée ici pour obtenir notre solution.

IV.3. Présentation du formalisme utilisé

IV.3.1. Les automates

Le formalisme des automates appartient à la catégorie des systèmes à événements discrets. Parmi ceux-ci, nous nous intéressons aux automates communicants qui sont une sous-classe des automates temporisés définis par (Alur and Dill, 1994). Ceux-là partagent des variables et sont synchronisés par l'ajout de labels / d'étiquettes sur les transitions.

Un automate communicant A est défini par un N-uplet $A = (Q, X, E, T, q_0, x_0)$ avec :

- Q , l'ensemble fini des localités ;
- X , l'ensemble fini des variables entières ;
- E , l'ensemble fini des étiquettes de transition composé de 3 ensembles disjoints :
 1. $L = L_e \cup L_r$, l'ensemble des labels de synchronisation avec L_e , l'ensemble des labels d'émission notés « label! » et L_r , l'ensemble des labels de réception notés « label? » ;
 2. G , l'ensemble des gardes (conditions booléennes sur les variables de X) ;
 3. M , l'ensemble des mises à jour (modification des valeurs des variables de X) ;
- $T \subset Q \times E \times Q$, l'ensemble des transitions. Celles-ci permettent de passer d'une localité à une autre (Q vers Q) en modifiant les valeurs des variables de X en fonction des étiquettes associées ;
- $q_0 \in Q$, la localité initiale ;
- $x_0 : X \leftarrow \mathbb{N}$, la valuation initiale des variables.

Nous pouvons aussi noter $S \subset Q \times X$ l'ensemble des états de l'automate. En effet, un état de l'automate est défini à tout instant par la localité active associée aux valeurs des variables au même instant.

Une trace d'exécution est une succession d'états de S et de transitions de T à partir de l'état initial

$$S_0 = (q_0, x_0) : S_0 \xrightarrow{t_1} S_1 \xrightarrow{t_2} S_2 \dots \xrightarrow{t_n} S_n$$

IV.3.2. Un réseau d'automates

Un réseau de n automates communicants $(NA = A^1 || A^2 || \dots || A^n)$ est défini par $NA = (Q, X, E, T, q_0, x_0)$ avec :

- $Q = Q^1 \cup Q^2 \cup \dots \cup Q^n$ qui est l'ensemble finis des localités ;
- $X = X^1 \cup X^2 \cup \dots \cup X^n$ qui est l'ensemble fini des variables entières ;
- $E = E^1 \cup E^2 \cup \dots \cup E^n$ qui est l'ensemble fini des étiquettes de transition ;
- $T = T^1 \times T^2 \times \dots \times T^n \subset Q \times E \times Q$ qui est l'ensemble des transitions ;
- $q_0 = q_0^1 \times q_0^2 \times \dots \times q_0^n$ qui définit l'ensemble des localités initiales ;
- $v_0 = v_0^1 \times v_0^2 \times \dots \times v_0^n : X \leftarrow \mathbb{N}$ qui définit la valuation initiale des variables.

Comme précédemment, un état du réseau est noté $S \subset Q \times X$ et cela correspond à l'ensemble des localités actives et les valeurs des variables à cet instant là.

De même, une trace d'exécution est une succession d'états de S et de transitions de T à partir de l'état

$$\text{initial : } S_0 \xrightarrow{t_1} S_1 \xrightarrow{t_2} S_2 \dots \xrightarrow{t_n} S_n$$

Une évolution du réseau d'automates conduisant d'un état S à un état S' est possible seulement dans un des deux cas suivants :

1. Une transition $t \in T$ peut être franchie dans un automate si la localité $q \in Q$ source de cette transition est active, que la garde $g \in G$ de l'étiquette $e \in E$ de cette transition est vraie et si cette étiquette e ne possède pas de label de synchronisation $l \in L$.
2. Deux transitions $(t_j, t_k) \in T^\alpha \times T^\beta$ d'un couple d'automates (A^α, A^β) sont franchies simultanément si leurs localités sources sont actives, si leurs gardes associées $(g_j, g_k) \in G^\alpha \times G^\beta$ sont vraies et si leurs étiquettes respectives $(e_j, e_k) \in E^\alpha \times E^\beta$ contiennent respectivement un label d'émission $l_j \in L_e^\alpha$ noté « $l_j!$ » et un label de réception $l_k \in L_r^\beta$ noté « $l_k!$ » dont le est identique ($l_k = l_j$).

Un exemple contenant ces deux types de transitions est présenté sur la Figure 9 en utilisant la formalisation graphique : les localités sont des cercles, les transitions des arcs orientés reliant deux localités, la localité initiale est désignée par un arc source, les gardes sont écrites entre crochets, les mises à jour sont soulignées et les labels d'émissions et de réceptions sont en italiques.

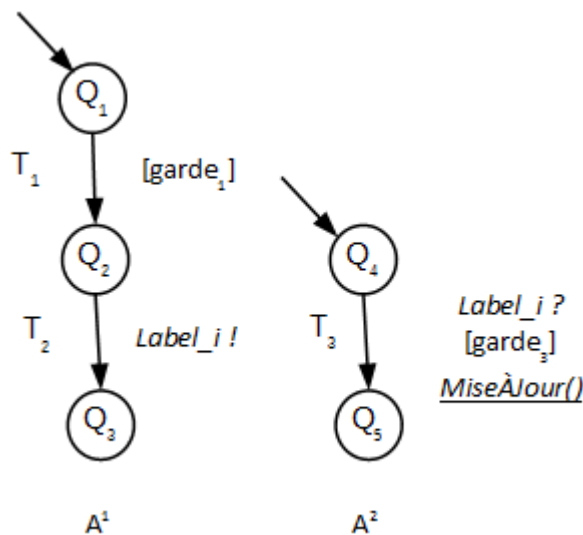


Figure 9: Exemple de l'évolution synchrone de deux automates communicants.

Cet exemple est composé d'un réseau de deux automates, A^1 et A^2 . Initialement, les deux localités actives sont Q_1 et Q_4 . La transition T_1 est franchissable d'après la première règle, T_2 n'est pas franchissable car sa localité source n'est pas active et T_3 n'est pas franchissable, car le label de réception qui lui est associé n'a pas d'équivalent parmi les transitions franchissables, la deuxième règle n'est pas respectée.

Une fois T_1 franchie, les localités actives sont Q_2 et Q_4 . Cette fois-ci, T_2 et T_3 respectent la règle 2 et le label d'émission/réception $label_i$ permet de franchir simultanément ces deux transitions. Une fois celles-ci franchies, les localités actives sont Q_3 et Q_5 . L'état du système est alors l'ensemble de ces localités actives, ainsi que les valeurs des variables X qui viennent d'être modifiées par la mise à jour $MiseÀJour()$.

IV.4. Modélisation du problème de rééquilibrage

IV.4.1. Les tâches

Le modèle générique TM d'une tâche t_i est défini par un automate avec trois localités et présenté ci-dessous et sur la Figure 10.

- Dans la première localité, la tâche t_i est en attente d'une allocation sur un poste de travail w_k . La transition qui peut être franchie correspond à l'émission d'une demande d'allocation envoyée à un poste de travail w_k .
- Une fois cette transition franchie, le modèle est en attente d'une réponse envoyée par le modèle de poste de travail (voir section suivante) qui peut être soit un refus, et dans ce cas la tâche retourne à sa localité initiale, prête pour une nouvelle demande (le délai entre une réponse et une nouvelle demande est instantanée, c'est la vérification des contraintes qui utilise du temps de calcul), soit une acceptation, et dans ce cas le modèle passe dans la dernière localité.
- La dernière localité du modèle représente une tâche allouée qui ne peut pas faire de nouvelle demande (unicité de l'allocation).

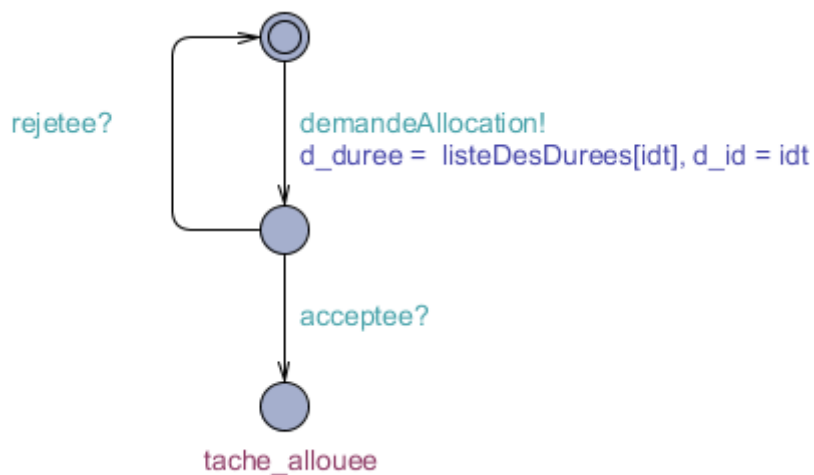


Figure 10: Modèle générique d'une tâche.

IV.4.2. Les postes de travail

Le modèle générique VM d'un poste de travail w_k est défini par un automate avec deux localités. Il est présenté ci-dessous et sur la Figure 11. Le poste va recevoir les demandes des tâches et répondre en fonction des contraintes définies précédemment (C1 et C2).

- Dans la première localité, le poste de travail w_k est en attente d'une demande d'allocation. Une fois la requête d'une tâche reçue, le modèle WM atteint la localité nommée « Calcul de la réponse ».
- Deux transitions partent de cette transition, mais les gardes, qui sont exclusives l'une par rapport à l'autre, ne permettent d'en franchir qu'une seule en fonction des contraintes :

si l'expression logique (C1 et C2) est fausse, le poste de travail émet un message de rejet de la demande et retourne dans sa localité initiale ;

si C1 et C2 sont vérifiées simultanément, le poste de travail accepte et envoie un message d'acceptation. Si l'allocation a été acceptée, la valeur $A(t_i) = w_k$ est enregistrée.

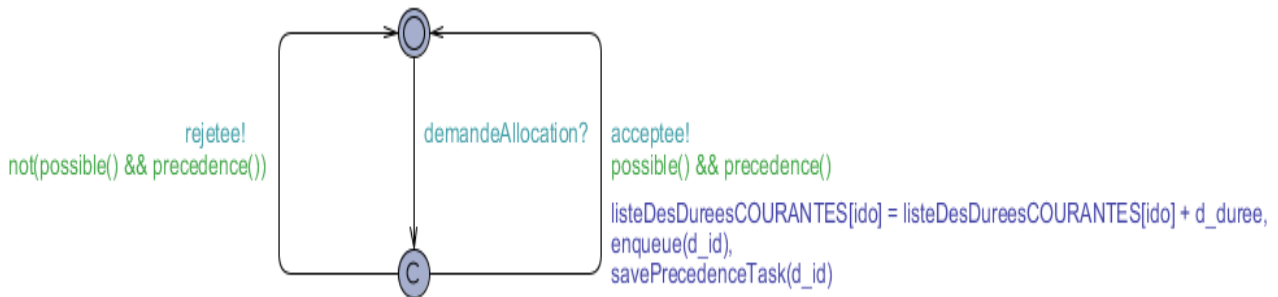


Figure 11: Modèle générique d'un poste de travail.

IV.4.3. La ligne d'assemblage

Le modèle complet de notre système est composé par un réseau formé d'instances des deux modèles précédents. (TM 1 || TM 2 || ... || TM $T_{\max}$ || WM 1 || WM 2 || ... || WM $W_{\max}$):

- $(T_{\max} - m)$ instances du modèle de tâches avec m le nombre de tâches déjà terminées lors du rééquilibrage,
- $W_{\max}$ instances du modèle de poste de travail (la capacité initiale des postes est adaptée pour tenir compte des tâches déjà terminées).

La synchronisation est assurée par l'envoi et la réception de messages (« demande d'allocation », « rejetée », « acceptée ») De plus, pour éviter qu'une tâche effectue une demande alors qu'une n'a pas encore reçu de réponse, la localité « calcul de la réponse » est verrouillée : une réponse doit être envoyée pour pouvoir quitter cette localité et pouvoir faire une nouvelle demande.

IV.5. La recherche d'atteignabilité

IV.5.1. Principe

Le modèle défini par le réseau d'automates représente alors toutes les réallocations possibles. Par construction, elles respectent les contraintes, cependant, elles ne sont pas toutes en accord avec l'objectif. La recherche d'atteignabilité sert alors à trouver une de ces solutions acceptables. Initialement, celle-ci est utilisée pour la vérifier formellement si un modèle peut atteindre certains états ou les éviter totalement. Elle fait donc partie des méthodes de vérification de modèles pour vérifier certains types de propriétés. Elle peut s'appliquer sur des réseaux d'automates temporisés (Subbiah and Engell, 2010) ou non, mais aussi sur des réseaux de petri (Schnoebelen et al., 1999).

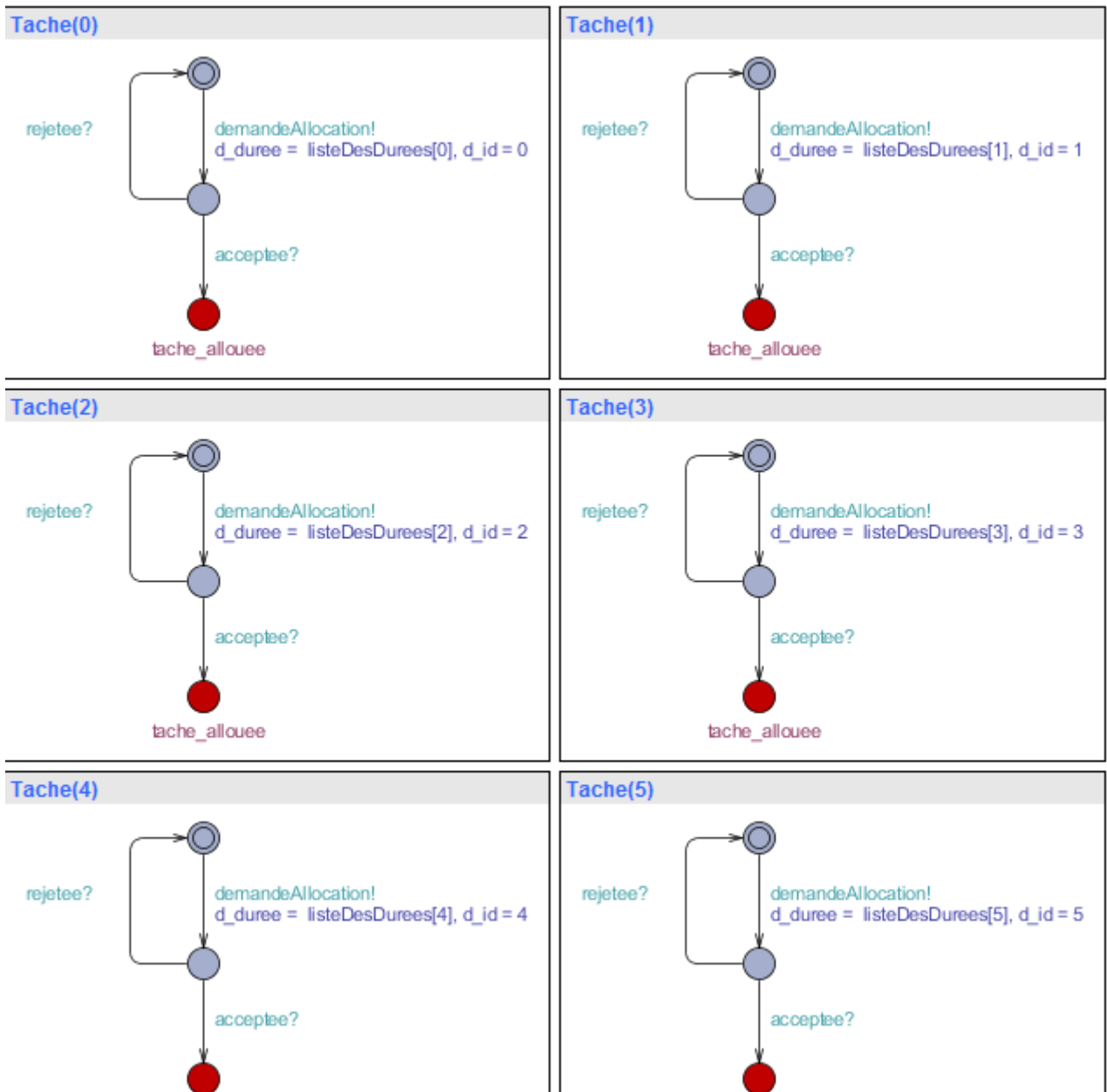


Figure 12: Modèle d'une ligne d'assemblage de 3 postes de travail et 6 tâches.

Cette vérification formelle est beaucoup utilisée dans la sûreté de fonctionnement afin de concevoir un système de commande fiable en s'assurant que les états indésirables ne peuvent pas être atteints et qu'en cas de panne, le système atteint un état de sécurité. Cette approche est donc valable, que le système à piloter soit un avion (Castel and Seguin, 2001), une centrale nucléaire ou bien un système manufacturier (Riera et al., 2012).

Le rôle d'un logiciel de model-checking est de vérifier qu'un modèle formel respecte diverses de ces propriétés formelles. Pour cela, le logiciel parcourt l'espace d'états du modèle et vérifie chaque propriété. Il peut, de plus, fournir un exemple ou un contre-exemple afin de compléter la réponse fournie (Berard et al., 2001).

L'avantage de cette méthode est de laisser le logiciel explorer l'espace d'états. Une fois le système modélisé, son comportement émerge par la composition des automates et la solution est trouvée par l'exploration de cet espace d'états.

IV.5.2. Définition de la propriété

Pour une recherche d'atteignabilité, la propriété (le problème) est de savoir si un état est atteignable, c'est à dire s'il existe un chemin (enchaînement de transitions) conduisant à l'état spécifié. On choisit donc une propriété que doit vérifier cet état. L'état sera ainsi une solution possible, et la trace fournie par l'outil sera la façon d'obtenir cette solution.

Ici la propriété formelle consiste à trouver s'il existe un état atteignable tel que l'allocation des tâches respecte l'objectif (les contraintes sont vérifiées à chaque étape du processus par les gardes des automates), c'est à dire : « Peut-on atteindre un état où toutes les tâches sont allouées et telles que le takt time soit respecté ? »

Une solution acceptable (respectant toutes les contraintes, mais pas forcément l'optimale) est obtenue s'il existe un état du système où toutes les tâches sont allouées de manière à ce que la durée de travail prévu sur chaque poste (D_w) respecte le takt time. La valeur des variables enregistrées pendant l'exécution de la trace ayant conduit à cet état forme ainsi la solution recherchée.

Cette propriété est exprimée en utilisant une logique non temporelle. Pour cela, on écrit la propriété suivante en utilisant CTL expression (Computation Tree Logic) (Clarke and Emerson, 1982) :

(P1) $EF(\text{« toutes les tâches sont allouées et le takt time respecté »}) = EF(\text{« toutes les tâches sont allouées et } W_i \leq W, d_{w(i)} \leq Takt\ Time \text{ »})$

avec E le quantificateur d'existence d'un chemin et F le quantificateur temporel de l'éventualité.

La propriété P1 signifie donc : existe-t-il un chemin tel que « toutes les tâches sont allouées et le takt time respecté » sera vraie un jour ?

V. Comparaison des 2 méthodes

L'objectif de cette partie est d'appliquer les 2 méthodes présentées sur un jeu d'instances provenant de l'entreprise. Après avoir présenté le contexte et les instances, nous les évaluerons à l'aide d'indicateurs de la littérature avant de terminer par la présentation des résultats obtenus pour chacune.

V.1. Présentation du contexte et des instances

L'objectif est de tester les méthodes sur 6 scénarios correspondant à 3 lignes d'assemblage de l'entreprise pour lesquelles on teste différentes perturbations survenant sur des postes différents.

Les instances issues de l'entreprise proviennent du feeder assemblant les sous-ensembles, les échangeurs de chaleur. Ceux-ci sont des éléments clés de l'entreprise car ils sont utilisés dans tous les produits assemblés par l'usine et leurs temps de fabrication sont aussi longs que l'assemblage des produits finaux. Ces contraintes associées au grand nombre de variantes font que, tout retard sur l'un des postes de ce feeder impacte la production des lignes principales : il faut donc maintenir l'équilibre de l'ensemble.

Ce feeder dispose d'un poste de travail constitué d'une enceinte de test spécifique rendant toutes les tâches de ce processus non déplaçables. Après ce poste de test, toutes les variantes sont regroupées car les tâches de finition effectuées après le test sont les mêmes pour tous. À l'inverse, avant le poste de test, les variantes sont séparées en deux lignes traitant chacune une famille de produits (Figure 13).

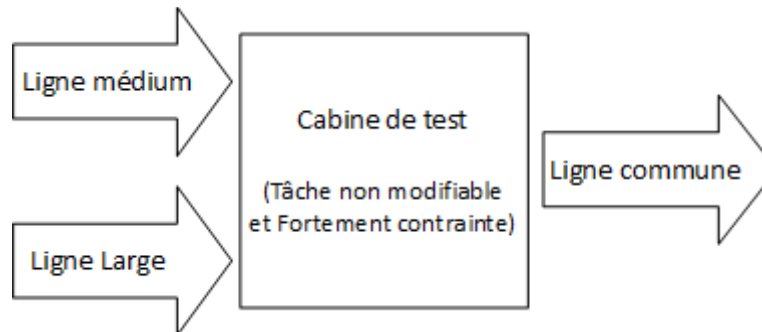


Figure 13: L'organisation du feeder échangeur.

Les processus de fabrication et le nombre de produits passant sur chaque ligne étant différents, les takts sont adaptés en conséquence. Nous pourrions donc en extraire 3 instances différentes.

De plus, cette ligne a pour particularité de mélanger des processus de fabrication, des opérations de montage ainsi que des tests. La variabilité entre les durées des tâches est donc importante et les contraintes de précédence sont importantes sur certaines parties de la ligne. Nous pourrions donc évaluer l'importance de la variabilité des tâches, l'impact du nombre de contraintes de précédence. Nous pourrions aussi nous affranchir des contraintes de précédence en ne considérant que les postes de montage et les tâches correspondantes.

Pour chaque ligne d'assemblage, le nombre de tâches et le nombre de postes sont définis par les deux ensembles T et W (voir formalisation, chapitre 1). La ligne est initialement définie par l'entreprise : équilibrage initial A_0 . (pour toutes les tâches t_i , $A_0(t_i)$ est défini) (Figure 14).

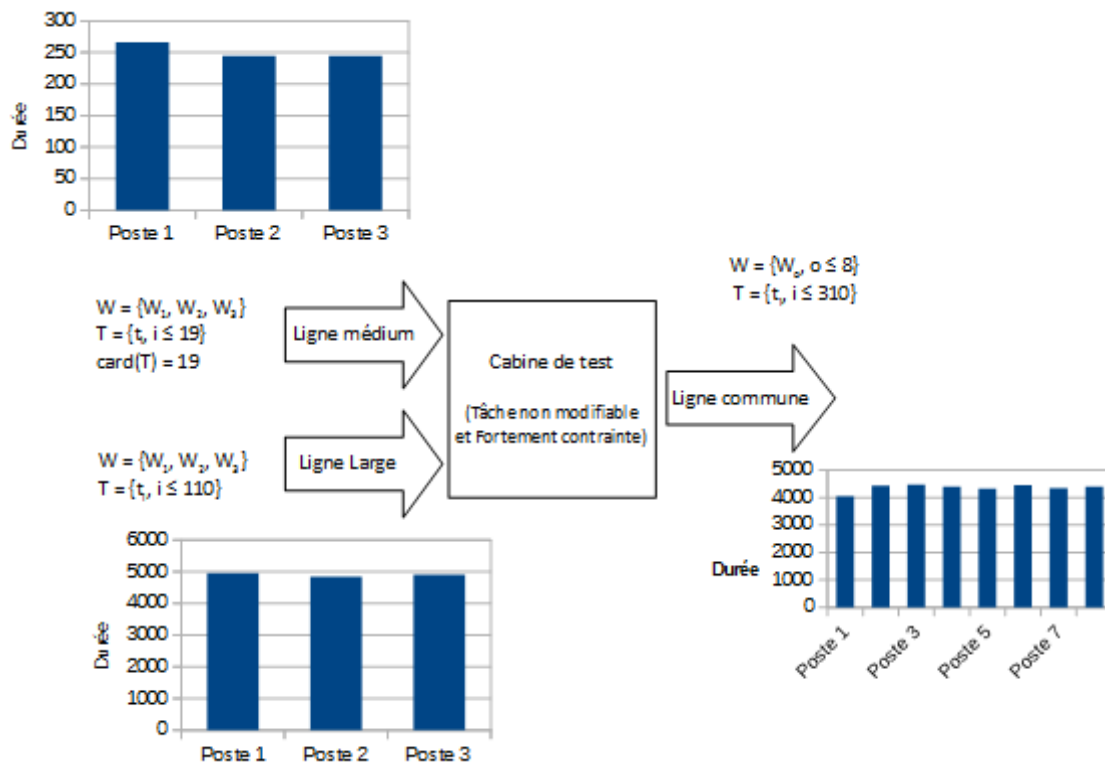


Figure 14: Équilibrage initial pour chacune des 3 instances.

Ensuite, pour chaque ligne on définit plusieurs instances. Une instance est spécifiée par un retard Δt survenant sur une tâche t_0 (pendant qu'elle est réalisée sur un poste $W_0 = A_0(t_0)$). Cela permet de définir les ensembles T_{bis} et W_{bis} qui sont les tâches et les postes qui peuvent être pris en compte pour la réallocation.

En effet, si le retard survient lors de la dernière tâche, la résolution sera totalement différente du cas où il survient au premier poste. Nous testons donc 2 aléas différents représentés par 2 scénarios différents incluant un retard sur le premier poste et un retard survenant sur un poste plus proche de la fin de la ligne).

	Ligne médium, scénario 1	Ligne large, scénario 2	Ligne commune, scénario 3	Ligne commune, scénario 4
Durée retard	40s	10s	100s	100s
Tâche en retard	t_4	t_{20}	t_{250}	t_{150}
Poste en retard	W_1	W_2	W_7	W_5
W_{bis}	$\{W_2, W_3\}$	$\{W_2, W_3\}$	$\{W_7, W_8\}$	$\{W_5, W_6, W_7, W_8\}$
$\text{card}(T_{bis})$	16	90	60	159
Takt time	266	5000	4500	4500

Tableau 1: Description des scénarios.

V.2. Évaluation des instances

V.2.1. Les indicateurs

Pour évaluer les instances, les indicateurs inspirés de la littérature sont décrits ci-dessous. Dans le cas du rééquilibrage dynamique, nous appliquons la distinction entre les tâches déjà réalisées avant la détection du retard et celles qui n'ont pas encore été effectuées et qui peuvent donc être réallouées. Par la suite, nous nous concentrerons donc sur ces tâches disponibles pour la réallocation.

Les principaux indicateurs utilisés pour définir la difficultés des instances sont présentes dans de nombreux articles comme par exemple (Schol, 1999). Ces instances ont une influence directe sur le nombre d'états possibles :

- Nombre de tâches de l'instance (dans notre cas on considère les tâches déplaçables (T_{bis})) ;
- Le nombre de postes de l'instance (dans notre cas les postes non bloqués (W_{bis})) ;
- Durée moyenne des tâches ;
- Variabilité des durées des tâches, rapport T_{max} sur T_{min} ;
- Nombre de contraintes de précédence.

En plus de ces indicateurs, nous pouvons utiliser des rapports de deux indicateurs. En effet, les effets des indicateurs sur la difficulté des instances ne sont pas indépendants :

- Rapport entre le nombre de contraintes par rapport au nombre de tâches ;
- Écart entre la moyenne des durées des tâches (temps de cycle) et le takt atteignable.

De plus, l'équilibrage initial a aussi un rôle dans la difficulté de l'instance. En effet, si le nombre de postes disponibles pour recevoir des allocations de tâches change, les indicateurs suivant sont modifiés aussi :

- La marge disponible par poste qui correspond à l'écart de durée entre la durée d'une opération (somme des durées des tâches allouées à ce poste) et le takt time. En effet, les marges des postes déjà terminés ne pourront pas être utilisées car appartenant au passé.
- La marge globale, qui est la somme des marges par poste, permet d'évaluer l'instance par rapport à la ligne. En effet, si les derniers postes sont plus chargés que les premiers, lors d'une reconfiguration, il sera plus difficile de trouver une solution acceptable, la marge disponible étant déjà passée.

V.2.2. Valeur des indicateurs

Dans le cas d'un rééquilibrage, nous étudions les tâches qui sont déplaçables : la marge disponible pour l'ensemble des postes varie donc d'une instance à l'autre. Nous pouvons notamment faire remarquer le cas de l'apparition d'un aléa sur le dernier poste de travail : le retard ne pourra pas être absorbé par une simple réallocation, il faudra un autre levier d'action comme la parallélisation, l'ajout d'un opérateur, etc.

Concernant les autres indicateurs, nous pouvons remarquer que dans notre cas, la répartition des durées des tâches n'est pas uniforme : il y a beaucoup de tâches de faibles durées et quelques singletons de grandes durées (ligne 3, 8 postes et 309 tâches) (Figure 15).

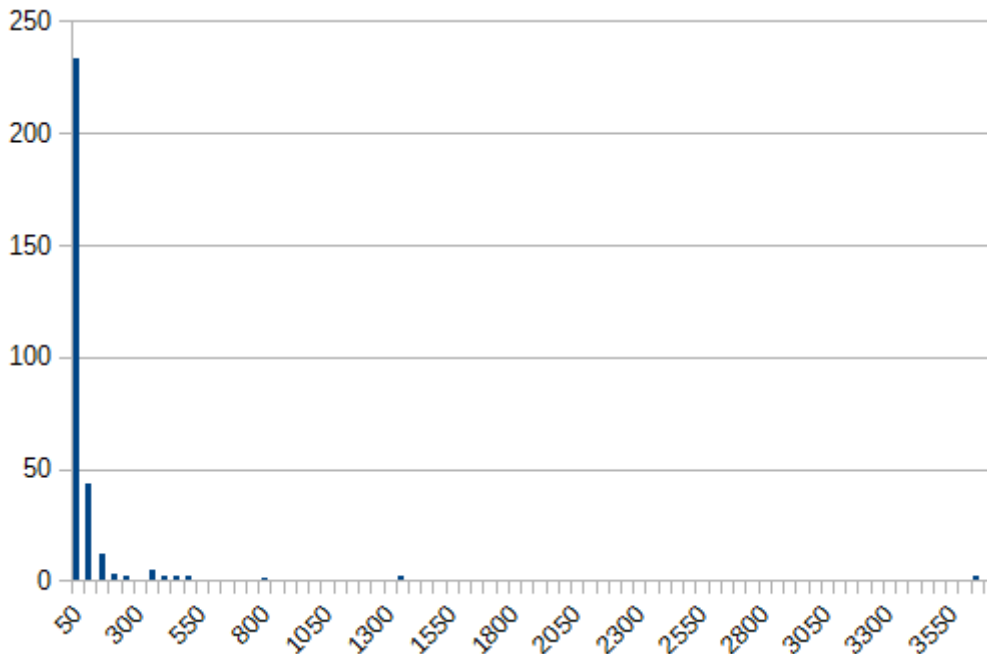


Figure 15: Répartition des durées des tâches de la ligne 1.

Cependant, et cela se retrouve dans d'autres articles de la littérature, cela est justifié par les contraintes industrielles. Les grandes tâches seront souvent dédiées à un poste sur lequel on a réservé du temps spécifiquement pour cette tâche insécable.

Ainsi l'exemple ci-dessous (Nasir Saidi Rashboogari) présente aussi un singleton.

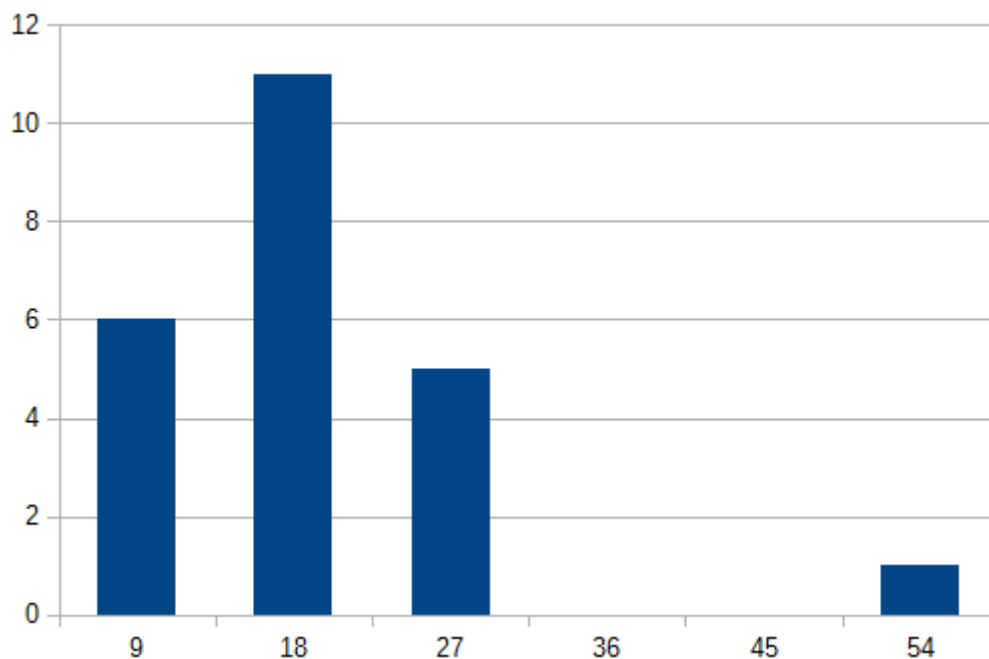


Figure 16: Répartition des durées des tâches de (Rashboogari and Kermanshah, 2012).

(Rashboogari and Kermanshah, 2012) ont utilisé l'instance de : « the precedence diagram of Kilbrid's benchmark problem, consists of 47 tasks and the value of cycle time is 56. »

Dans d'autres instances, on peut trouver une répartition proche d'une loi normale comme dans (Dolgui et al., 2005).

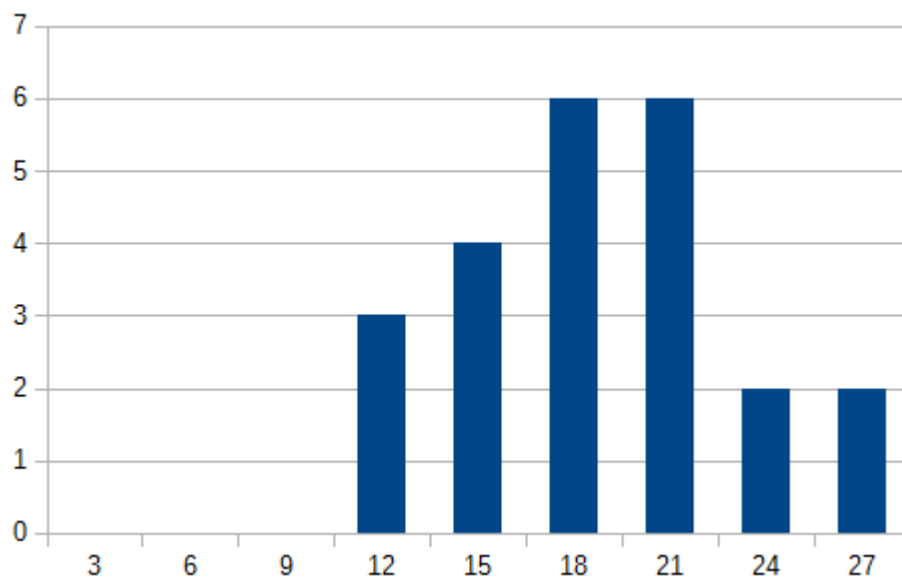


Figure 17: Répartition des durées des tâches de (Dolgui, 2005).

Le Tableau 2 ci-dessous présente les valeurs des indicateurs définis dans la partie précédente. Ces valeurs proviennent des cas d'étude examinés dans ce chapitre.

	Scénario 1	Scénario 2	Scénario 3	Scénario 4
W_{bis}	$\{W_2, W_3\}$	$\{W_2, W_3\}$	$\{W_7, W_8\}$	$\{W_5, W_6, W_7, W_8\}$
$card(T_{bis})$	16	90	60	159
Durée moyenne (en s)	38	128	105	93
Durée minimale (en s)	20	15	15	15
Durée maximale (en s)	60	3660	840	840
T_{max} / T_{min}	3	244	56	56
Marge globale (en s)	42	318	17	242

Tableau 2: Valeur des indicateurs pour les différents scénarios.

Sans que cela soit une nécessité compte tenu de nos obligations industrielles, nous pouvons donc noter que par rapport à la littérature, notre ensemble de tâches est donc cohérent.

V.3. Évaluation des résultats obtenus

V.3.1. Les éléments d'évaluation

Dans le cadre du rééquilibrage dynamique, la vitesse de résolution est plus importante que l'optimalité de la solution obtenue. C'est pourquoi nous ne relevons pas le coefficient du lissage obtenu (ni la marge par poste présentée précédemment pour l'équilibrage initial). Nous évaluons donc :

- Le temps de cycle trouvé (celui-ci peut servir à des rééquilibrages plus globaux pour les prochains calculs prédictifs). Il s'agit du temps du poste le plus chargé.
- Le temps de résolution
- Le nombre de tâches réallouées par rapport l'équilibrage initial (nombre de changements). Cet indicateur peut être rapproché du critère de ré-utilisabilité proposé par (Ko et al., 2005) qui est le rapport entre les ressources utilisées pour la nouvelle ligne et les ressources réellement disponibles.

Pour chacun des 4 scénarios, plusieurs résolutions ont été effectuées. En effet, la méthode fournissant une solution acceptable et ne cherchant pas l'optimalité, diverses solutions existent. Une moyenne a donc été effectuée. Nous notons que la variance des résultats autour de cette moyenne aurait été tout aussi pertinent.

Le temps de cycle est défini comme le temps maximal des durées des postes de travail. Une solution acceptable est une solution dont le temps de cycle est inférieur au takt de rééquilibrage. Ce takt de rééquilibrage est calculé à partir du takt initial corrigé par une marge de tolérance permettant d'absorber le retard si la charge de travail globale est trop importante.

V.3.2. Scénario 1 :

- takt de recalcul pour les postes de $W_{bis} = 266$: ce takt de recalcul correspond au takt initial, cependant, afin de pouvoir absorber le retard et la variabilité intrinsèque du processus physique, une marge d'erreur epsilon est tolérée de 25 s.

	Nb de changements	Temps en s.	Temps de cycle
Test 1	8	Arrêt à 30s	275
Test 2	10	Arrêt à 30s	272
Test 3	10	Arrêt à 30s	275
Test 4	4	Arrêt à 30s	281
Test 5	10	Arrêt à 30s	286
Test 6	11	Arrêt à 30s	287
Test 7	11	Arrêt à 30s	285
Test 8	8	Arrêt à 30s	286
Test 9	12	Arrêt à 30s	281
Test 10	10	Arrêt à 30s	275
Test 11	14	Arrêt à 30s	272
Moyenne	9,818	Arrêt à 30s	279,545

Tableau 3: Résultats de la méthode ILS pour le scénario 1.

	Nb de changements	Temps en s.	Temps de cycle
Test 1	3	0,015	285
Test 2	11	0,219	287
Test 3	3	0,046	291
Test 4	2	0,031	280
Test 5	11	3,822	275
Test 6	6	36,473	284
Test 7	2	0,109	276
Test 8	1	0,031	279
Test 9	2	0,016	280
Test 10	12	0,046	282
Moyenne	5,4	4,0808	281,9

Tableau 4: Résultats de la recherche d'atteignabilité pour le scénario 1.

Cependant, lors de la recherche de la solution et la réalisation des tests ci-dessus, si la solution n'est pas trouvée au bout d'un certain temps (30s), un script stoppe l'exploration pour relancer celle-ci sur une autre

zone de l'espace de solutions. Dans ces cas-là, le temps de résolution augmente de 30 secondes, on obtient alors les données corrigées suivantes :

	Nb de changements	Temps en s.	Temps de cycle
Test 1	3	0,015	285
Test 2	11	0,219	287
Test 3	3	0,046	291
Test 4	2	0,031	280
Test 5	11	3,822	275
Test 6	6	36,473	284
Test 7	2	0,109	276
Test 8	1	30,031	279
Test 9	2	60,016	280
Test 10	12	30,046	282
Moyenne	5,4	16,0808	281,9

Tableau 5: Résultats de la recherche d'atteignabilité pour le scénario 1.

V.3.3. Scénario 2 :

- takt de recalcul pour les postes de $W_{bis} = 5000$

	Nb de changements	Temps en s.	Temps de cycle
Test 1	48	2	4970
Test 2	55	2	4949
Test 3	50	1	4965
Test 4	71	3	4950
Test 5	42	1	4935
Test 6	59	2	4965
Test 7	38	<1	4994
Test 8	63	2	4995
Test 9	63	2	4925
Test 10	55	2	4980
Moyenne	54,4	2	4963,8

Tableau 6: Résultats de la méthode ILS pour le scénario 2.

	Nb de changements	Temps en s.	Temps de cycle
Test 1	39	13,962	5059
Test 2	47	0,515	4939
Test 3		Arrêt après 1min	
Test 4	55	9,672	4950
Test 5	50	122,475	4950
Test 6	38	0,483	4944
Test 7	43	0,375	4935
Test 8	53	0,452	4935
Test 9		Arrêt après 1min	
Test 10		Arrêt après 1min	
Test 11	45	0,358	5029
Test 12	45	0,702	4939
Test 13		Arrêt après 1min	
Test 14	50	0,608	4945
Moyenne	46,5	14,9602	4962,5

Tableau 7: Résultats de la recherche d'atteignabilité pour le scénario 2.

V.3.4. Scénario 3 :

- takt de recalcul pour les postes de $W_{bis} = 4500$

	Nb de changements	Temps en s.	Temps de cycle
Test 1	9	Arrêt après 1min	4985
Test 2	9	<1	4980
Test 3	10	<1	4980
Test 4	7	<1	4980
Test 5	9	Arrêt après 1min	4985
Test 6	8	Arrêt après 1min	5045
Test 7	8	<1	4980
Test 8	4	<1	4980
Test 9	8	Arrêt après 1min	4985
Test 10	11	Arrêt après 1min	4985
Test 11	11	Arrêt après 1min	4985
Test 12	8	<1	4980
Test 13	9	<1	4980
Moyenne	8,538	28	4986,923

Tableau 8: Résultats de la méthode ILS pour le scénario 3.

	Nb de changements	Temps en s.	Temps de cycle
Test 1	3	0,258	4974
Test 2	5	0,376	4989
Test 3	2	0,301	4944
Test 4	3	0,324	5004
Test 5	4	0,344	5019
Test 6	6	0,671	4980
Test 7	5	0,299	4930
Test 8	4	0,551	4945
Moyenne	4	0,39	4973,125

Tableau 9: Résultats de la recherche d'atteignabilité pour le scénario 3.

Dans le tableau 9, de même que dans le tableau 4, seuls les tests ayant abouti à une solution ont été chronométré. Lorsque la solution n'est pas trouvée avant la fin d'un délai spécifié, un script stoppe l'exploration pour relancer celle-ci sur une autre zone de l'espace de solutions. Pour ce troisième scénario, cette relance a eue lieu 33 fois : cela a donc impacté la rapidité de résolution, mais la solution est finalement trouvée. De même, la méthode ILS n'a pas trouvé de solution respectant le takt, elle a donc

donné une solution pour laquelle le takt n'est pas respecté, mais néanmoins plus proche que l'équilibrage initial : la réallocation absorbe donc mieux le retard que l'absence d'action.

V.3.5. Scénario 4 :

	Nb de changements	Temps en s.	Temps de cycle
Test 1	6	2,447	4462
Test 2	1	1,403	4462
Test 3	1	1445	4462
Test 4	3	1,664	4462
Test 5	13	4,177	4462
Test 6	1	1,691	4462
Moyenne	4,166666	2,14	4462

Tableau 10: Résultats de la recherche d'atteignabilité pour le scénario 4.

Pour ce scénario, l'espace d'états est grand et l'exploration n'a pas abouti 25 fois sur les 30 essais effectués. Le script de résolution stoppant la recherche au bout de 30s avant de la relancer, la prise en compte de cette durée augmente le temps moyen de 125 secondes.

V.4. Analyse des résultats

Le nombre de changements entre l'équilibrage initial et la nouvelle allocation permet de connaître le nombre d'informations que l'on doit transférer aux opérateurs. Plus celui-ci est faible, plus l'application dans l'usine est simple et moins sujette aux erreurs. Ce nombre pourra aussi être interprété comme le coût de reconfiguration.

Nous remarquons alors que le nombre de ces changements est similaire entre ILS et la recherche d'atteignabilité.

Contrairement à la méthode ILS, si une solution acceptable n'est pas trouvée, nous ne disposons pas de solution dégradée. Si aucune solution n'est trouvée, il faut relancer le logiciel en élargissant la marge de tolérance autour du takt dans la propriété de la recherche d'atteignabilité. Cela est effectué par un script python qui relance l'exploration sur une zone différente de l'espace d'états.

Dans les deux cas, le temps de résolution est inférieur à la minute, même en prenant en compte le temps mis par le script pour tester différentes zones de l'espace d'états en relançant plusieurs fois la résolution. Donc cela est compatible avec notre besoin de réactivité.

Néanmoins, nous remarquons que plus le problème est grand (nombre de tâches à réallouer et nombre de postes disponibles), plus la résolution est nécessaire d'être relancée. Dans les cas extrêmes, la recherche d'atteignabilité explore l'espace d'états sans trouver une solution acceptable. Ainsi un retard appliqué sur la ligne 3, sur la tâche 64, au niveau du poste 3 n'aboutit pas à une solution ($\text{card}(\text{Tbis}) = 245$ et $\text{card}(\text{Wbis}) = 6$).

VI. Conclusion

Ces deux méthodes sont donc compatibles avec la notion de calcul de rééquilibrage dynamique (non optimal mais rapide). Cependant, le seul levier considéré dans cette étude est le déplacement des tâches. Pour changer les leviers d'actions pris en compte et intégrer la parallélisation des tâches ou le chevauchement par exemple, il faut alors redéfinir les différentes contraintes du problème. En effet, même s'il est très simple de changer le nombre de tâches, le nombre de postes ou bien encore l'équilibrage initial, il est plus compliqué de modifier les leviers d'action (façon de modifier l'allocation des tâches) que l'on souhaite intégrer dans l'algorithme d'équilibrage.

Dans les deux cas, la modification des caractéristiques de la ligne et des différents degrés de liberté à disposition pour procéder à un rééquilibrage nécessitent de repenser en profondeur le modèle et les algorithmes de rééquilibrage. Cette adaptation induit donc un coût et un temps de modélisation important non compatible avec le niveau de réactivité souhaité. L'objectif est donc de développer une approche de modélisation modulaire, basée sur l'utilisation d'éléments de résolution génériques, de telle sorte que l'adaptation des algorithmes de rééquilibrage se limite, dans la mesure du possible, à une instanciation et une paramétrisation d'éléments de résolutions génériques pré-définis.

Le chapitre suivant propose une approche originale basée sur l'utilisation des automates communicants et des algorithmes de recherche d'atteignabilité pour résoudre cette problématique de généricité.

Chapitre 3 : Proposition d'une méthodologie modulaire pour la résolution des ALBP

Table des matières

I. Introduction.....	71
II. Présentation générale de la méthodologie.....	71
III. Classe de problème : Définition d'une bibliothèque d'opérateurs élémentaires génériques.....	73
III.1. Présentation.....	73
III.2. Analyse fonctionnelle.....	74
III.3. Conception des modules réalisant les leviers d'action.....	76
<i>III.3.1. Conception de l'architecture des modules.....</i>	<i>76</i>
<i>III.3.2. Conception technique des modules.....</i>	<i>77</i>
<i>a. Le modèle générique d'un module maître.....</i>	<i>78</i>
<i>b. Le motif d'un module d'action.....</i>	<i>79</i>
<i>c. c. Le motif d'un module de vérification.....</i>	<i>79</i>
III.4. Implémentation des modules avec UPPAAL.....	80
<i>III.4.1. Choix du logiciel.....</i>	<i>80</i>
<i>III.4.2. Implémentation des modules.....</i>	<i>81</i>
IV. Problème spécifique : Instanciation des opérateurs génériques.....	81
IV.1. Choix des leviers et de la stratégie de reconfiguration.....	81
IV.2. Conception : assemblage des modules techniques.....	82
IV.3. Implémentation des modules avec le logiciel UPPAAL.....	83
<i>IV.3.1. Implémentation.....</i>	<i>83</i>
<i>a. Implémentation des modules.....</i>	<i>83</i>
<i>b. Implémentation de la propriété.....</i>	<i>84</i>
<i>c. Implémentation de la stratégie globale.....</i>	<i>84</i>
V. Conclusion.....	85

I. Introduction

Ce chapitre présente une méthodologie de modélisation à base d'automates communicants pour la résolution des problèmes de rééquilibrage de ligne d'assemblage. L'originalité de l'approche réside dans la définition a priori d'un ensemble d'opérateurs de résolution élémentaires et génériques qui seront ensuite, instanciés, paramétrés et assemblés au sein d'un script en tenant compte des caractéristiques de la ligne à rééquilibrer et des leviers d'actions à disposition.

La contribution scientifique repose donc sur deux points. D'une part, il s'agit de construire une bibliothèque d'opérateurs élémentaires de résolution d'un problème de rééquilibrage. La définition des opérateurs ainsi que de leurs paramètres s'appuient sur une démarche classique d'analyse, de conception puis d'implémentation. Dans un deuxième temps, il s'agit de proposer une démarche d'instanciation, de paramétrisation et d'assemblage des opérateurs présents en bibliothèque pour construire un algorithme de rééquilibrage spécifique au problème posé et à ses degrés de liberté.

II. Présentation générale de la méthodologie

La mise en œuvre d'une approche méthodologique, dont la nécessité a été expliquée précédemment, passe par une première étape majeure : la définition des étapes de la méthode. Ensuite et en s'appuyant sur les outils méthodes du génie logiciel et des travaux sur les méthodes de modélisation des modèles de simulation SED (Fleury et al., 1999 ; El-Haouzi, 2008 ; Gourgand et al., 2014)), nous proposons de diviser l'espace du problème en deux niveaux d'abstraction. Le premier niveau concerne la classe de problème considérée et le second, le problème spécifique à résoudre. Cela commence par la génération d'une librairie de modules qui sont ensuite instanciés afin de résoudre chaque problème spécifique. Ces deux parties sont représentées sur la figure suivante (Figure 18). Notons de plus, que si un module s'avère manquant pour traiter un problème spécifique, il est possible d'enrichir la librairie.

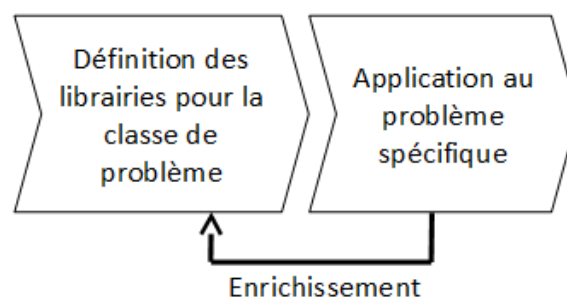


Figure 18 : Les étapes principales de la méthode proposée.

Chacune de ces deux parties peut être décomposée en trois phases comme présenté sur la Figure 19 :

- la phase d'analyse : une analyse fonctionnelle permettant de mettre en lumière les objectifs, les services à rendre par le modèle et les différents leviers d'action possibles pour les atteindre ;
- la phase de modélisation conceptuelle : Il s'agit de l'une des étapes préliminaires au développement d'une application. Le modèle conceptuel permet de représenter, éventuellement graphiquement, un ensemble de concepts liés sémantiquement entre eux, les relations entre les concepts étant décrites par des termes. Le modèle conceptuel appliqué à la conception d'une

application permet de décrire les processus logiques régissant le fonctionnement de l'application. Appliqué à la modélisation des données, le modèle conceptuel permet de représenter la hiérarchie des types de données, les caractéristiques des données et la nature des relations. Une modélisation est une forme de communication entre personnes alors que la spécification d'un modèle tend à être plus formelle et fait appel à la rigueur des mathématiques. Il existe plusieurs normes permettant de décrire un modèle conceptuel tel que : les modules de haut niveau et leur décomposition en modules techniques sont définis afin de réaliser chacune des fonctions définies dans l'étape précédente ;

- la phase d'implémentation où les modules définis précédemment sont implémentés avec le logiciel choisi.

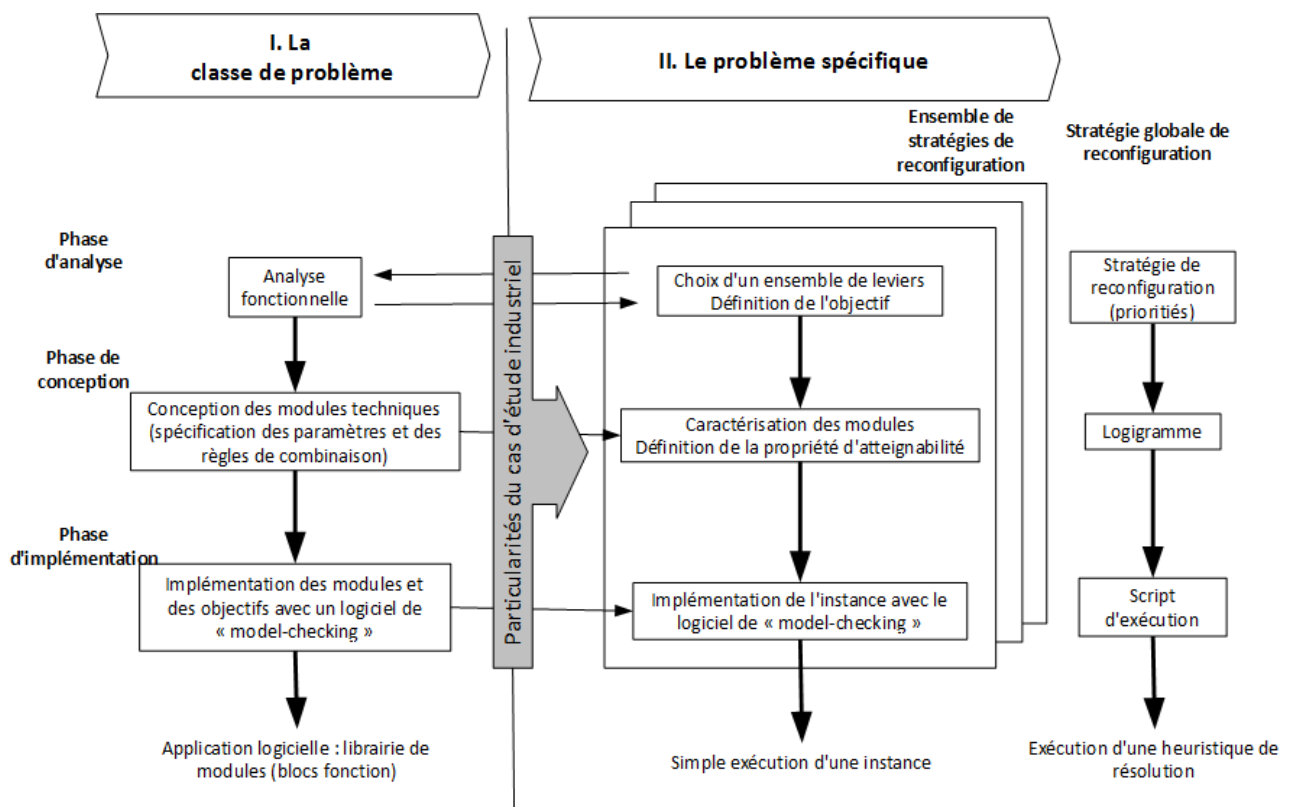


Figure 19: Les étapes de notre proposition méthodologique.

Le reste de ce chapitre présentera dans le détail chaque étape en commençant par l'approche de la classe de problèmes permettant la réalisation de la librairie en trois étapes. Les parties suivantes présenteront comment ces modules sont choisis, instanciés et implémentés pour résoudre chaque problème spécifique.

III. Classe de problème : Définition d'une bibliothèque d'opérateurs élémentaires génériques

III.1. Présentation

Cette partie a pour but de présenter les étapes de la méthode permettant d'obtenir une bibliothèque d'opérateurs élémentaires génériques pour la résolution de différents problèmes de rééquilibrage. Ces éléments du modèle doivent donc être réutilisables.

(Morand, 1995) propose une méthode pour réutiliser les solutions précédemment trouvées. Au fur et à mesure que des solutions aux problèmes sont élaborées, celles-ci sont testées pour résoudre des problèmes similaires. Sur ces nouvelles applications, seuls certains éléments peuvent être adaptés et la solution de ce nouveau problème est trouvée par modification, combinaison et adaptation des anciennes solutions. Lorsque la solution est utilisée avec succès pour résoudre plusieurs problèmes, celle-ci est acceptée, généralisée et normalisée.

Afin que ces éléments de modélisation génériques et réutilisables soient issus du savoir-faire métier, des travaux proposent d'utiliser un méta-modèle des concepts métiers nécessaires à la résolution de chaque modèle particulier. Ces concepts sont ensuite formalisés sous la forme d'une bibliothèque ou une collection de solutions génériques, chaque concept formant alors un élément de modélisation instanciable et utilisables pour résoudre un problème appartenant à un certain domaine. (Vernadat, 1996) définit cet ensemble comme un ensemble de « constructs » : « *a primitive of a modelling language, the syntax and semantics of which must be precisely defined. Formal description techniques are defined by constructs having a good syntax and semantics* ». Pour compléter cette définition, la norme (ENV-12204, 1996) propose une approche pour normaliser ces différents concepts métiers. Ainsi, ils définissent les « constructs » comme « *A textual or graphical artefact devised to represent in an orderly way the diverse information on common properties and elements of a collection of phenomena* ». Cela permet de structurer ces concepts : « *A grouping of constructs and rules for valid groupings which define the syntax of the model* ».

En parallèle, (Alexander et al., 1977) définit le concept de patron de modélisation comme une méthode pour résoudre « *un problème qui se manifeste constamment dans notre environnement, et qui décrit le cœur de la solution de ce problème, d'une façon telle que l'on puisse réutiliser cette solution des millions de fois sans jamais le faire deux fois de la même manière* ». Dans la prolongation de ces travaux, (Rieu et al., 1999) puis (Gzara, 2000) identifient la nécessité de capitaliser les connaissances métiers lors de la conception des patrons de modélisation. Pour cela, ils proposent de considérer chaque patron dans son contexte donné, puis proposent la mise en place de mécanismes de raffinement.

Ce type d'approche permettant de garantir la qualité des éléments de modélisation, et donc de garantir, à priori, la qualité d'un modèle réalisé à partir de ces éléments, nous semble intéressante. Elle sera donc suivie pour procéder à la définition d'opérateurs (patrons) génériques pour la résolution de problèmes de rééquilibrage. Plus précisément, la méthodologie proposée s'appuiera sur des démarches de modélisation système classiques (Frachet, 1987 ; Calvez, 1990 ; Morel, 1992 ; Denis, 1994 ; Meinadier, 2002) qui préconisent l'élaboration des modèles d'architectures :

- **structurels** décrivant l'agencement des fonctions et des modules logiques ainsi que la sémantique des flux informationnels et physiques qu'ils transforment ou s'échangent,
- **comportementaux** décrivant l'enchaînement des éléments logiques identifiés dans les modèles structurels ainsi que les réactions de ces éléments logiques (élaboration des variables de sortie) en réponse aux stimulus auxquels ils sont soumis,
- **informationnels** décrivant les structures de données sur lesquelles s'appuient les fonctions et constituants du système,

La ré-utilisabilité de la bibliothèque d'opérateurs de résolution est un élément clé de la méthodologie que nous proposons. Si la réutilisation est actuellement assez bien traitée au niveau de l'implémentation (bibliothèques de composants logiciels), elle reste très limitée au niveau de la conception, voire inexistante au niveau de l'analyse. Le cadre méthodologique que nous proposons offre donc des éléments pour permettre la réutilisation des opérateurs de résolution sur toutes les phases de conception du modèle. Ces opérateurs seront ainsi associés à des leviers d'actions mais ils pourront également être recombinaés pour créer de nouveaux leviers.

Un levier d'action est un moyen, choisi par l'entreprise, pour modifier l'équilibrage de la ligne d'assemblage. Le levier d'action insiste sur la fonction que l'on souhaite réaliser plutôt que la description des éléments physiques du système, c'est pourquoi une analyse fonctionnelle est privilégiée pour l'élaboration des patrons de modélisation.

III.2. Analyse fonctionnelle

La première étape a pour but d'identifier les objectifs des problèmes de rééquilibrage dynamique et les moyens de les atteindre. Pour cela une analyse fonctionnelle de la classe de problèmes est réalisée.

La première étape consiste à définir les fonctions principales ou de service de l'ensemble des rééquilibrages possibles. Ensuite celles-ci seront décomposées en fonctions techniques, lesquelles seront réalisées par les différents modules formant l'outil de résolution.

Pour cela à chaque niveau de l'analyse fonctionnelle, trois questions sont étudiées :

- À qui le système rend-il service ?
- Dans quel but agit le système ?
- Sur quoi agit le système ?

Dans le cas étudié dans cette thèse, le but est de concevoir un outil pour simplifier la résolution de différents rééquilibrages dynamiques en s'adaptant aisément à leurs contraintes spécifiques. Il rend donc service lorsqu'un rééquilibrage doit être recalculé.

La principale fonction que doit donc remplir le système de résolution est de trouver une nouvelle configuration de la ligne d'assemblage qui réalise l'objectif de lissage de charge tout en respectant les différentes contraintes.

Pour cela, le système de résolution dispose de plusieurs manières de modifier l'équilibrage de la ligne qui sont nommées les leviers d'action. En effet, pour modifier la répartition de la charge le long de la ligne d'assemblage (et donc de se rapprocher ou non de l'équilibrage désiré) nous pouvons changer le séquençement des produits assemblés sur la ligne ou bien changer l'allocation des tâches. Ces leviers s'appliquent sur différents éléments du système comme par exemple : produits (séquencement...), ressources (assignation des ressources...), postes de travail (nombre de postes de travail...), tâches (allocation des tâches...) (F1 F2 F3 F4).

De plus, pour définir complètement un levier d'action et s'adapter au problème spécifique, nous définissons, pour chaque levier, différents paramètres. Ces paramètres restreignent le champ d'application des leviers en posant des limites sur les différents éléments du système, ou sur l'action du levier (dernière colonne du Tableau 11). Ainsi, pour l'exemple, nous faisons la distinction entre le déplacement d'une tâche d'un poste vers le poste voisin et le déplacement d'une tâche vers n'importe quel autre poste. Cela englobe donc l'espace de départ d'un levier, son espace d'arrivée et son espace de travail.

Fonctions	Sous-fonctions, leviers d'action	Champs d'application des leviers
F4. Modifier l'allocation des tâches	F4.1. Déplacer une tâche (déplacement)	D'un poste vers un poste voisin
		D'un poste vers un autre
	F4.2. Exécuter une tâche en parallèle d'une autre (parallélisation)	D'un même poste de travail
		Qui ne sont pas sur un même poste de travail initialement
	F4.3. Exécuter une tâche partiellement en parallèle d'une autre (chevauchement)	D'un même poste de travail
		Qui ne sont pas sur un même poste de travail initialement

Tableau 11: Extrait du résultat de l'analyse fonctionnelle.

Si un problème de rééquilibrage nécessite un nouveau levier, celui-ci peut être ajouté à la liste. Les autres étapes de l'analyse sont alors réalisées pour définir les éléments sur lesquels il s'applique, ces restrictions et ainsi compléter la librairie. Cette amélioration peut aussi venir d'une précision de certains éléments existants (ou de restrictions dans la fonction, plutôt que ajout de nouveaux éléments).

À titre d'exemple, seulement la quatrième fonction sera détaillée : le modèle présenté ici ne travaillera que sur la réallocation des tâches sur les postes de travail de la ligne d'assemblage. En détaillant plusieurs possibilités pour déplacer les tâches, la décomposition suivante est obtenue (fonctions principales et fonctions techniques). Les autres fonctions devront être conçues et implémentées selon la même méthode.

Cet ensemble de sous-fonctions associées à leurs caractérisations (critères limitant le degré de leur réalisation) correspond à un ensemble de levier d'action permettant de modifier l'allocation des tâches afin de résoudre le problème de rééquilibrage de ligne d'assemblage. Les leviers à activer seront choisis parmi

cet ensemble en fonction des contraintes et des spécificités du problème à résoudre. Chacune de ses fonctions est réalisée par un ensemble de modules comme expliqué dans la partie suivante.

III.3. Conception des modules réalisant les leviers d'action

III.3.1. Conception de l'architecture des modules

Chacun des leviers d'action est représenté par un module de haut niveau qui est une composition de plusieurs modules élémentaires, lesquels, sont de trois types :

- **V** est l'ensemble des modules de vérification. Chacun d'eux est dédié à la vérification d'une contrainte particulière ;
- **A** est l'ensemble des modules d'action élémentaires. Chacun d'eux permet de réaliser une action élémentaire modifiant l'équilibrage de la ligne ;
- **M** est l'ensemble des modules maîtres. Chacun d'eux est dédié à un levier d'action et permet de coordonner le travail des autres modules comme décrit ci-dessous.

La composition des modules pour réaliser les leviers consiste à sélectionner (Figure 20) :

- 1 à n modules de vérification issu(s) de **V** ;
- 1 à n modules d'action issu(s) de **A** ;
- un et un seul module maître issu de **M**.

De plus, les leviers élémentaires de vérification et d'action peuvent être communs à plusieurs leviers. Ils seront alors appelés successivement par différents modules maîtres.

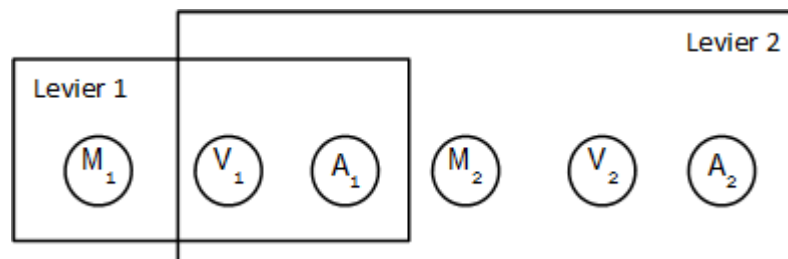


Figure 20: Composition de modules formant les leviers d'action.

L'interaction entre ces modules est réalisée par un mécanisme d'appels/réponses initié par chaque module maître. La Figure 21 illustre l'interaction entre les modules pour réaliser un levier d'action ne nécessitant qu'un seul module d'action et un seul module de vérification.

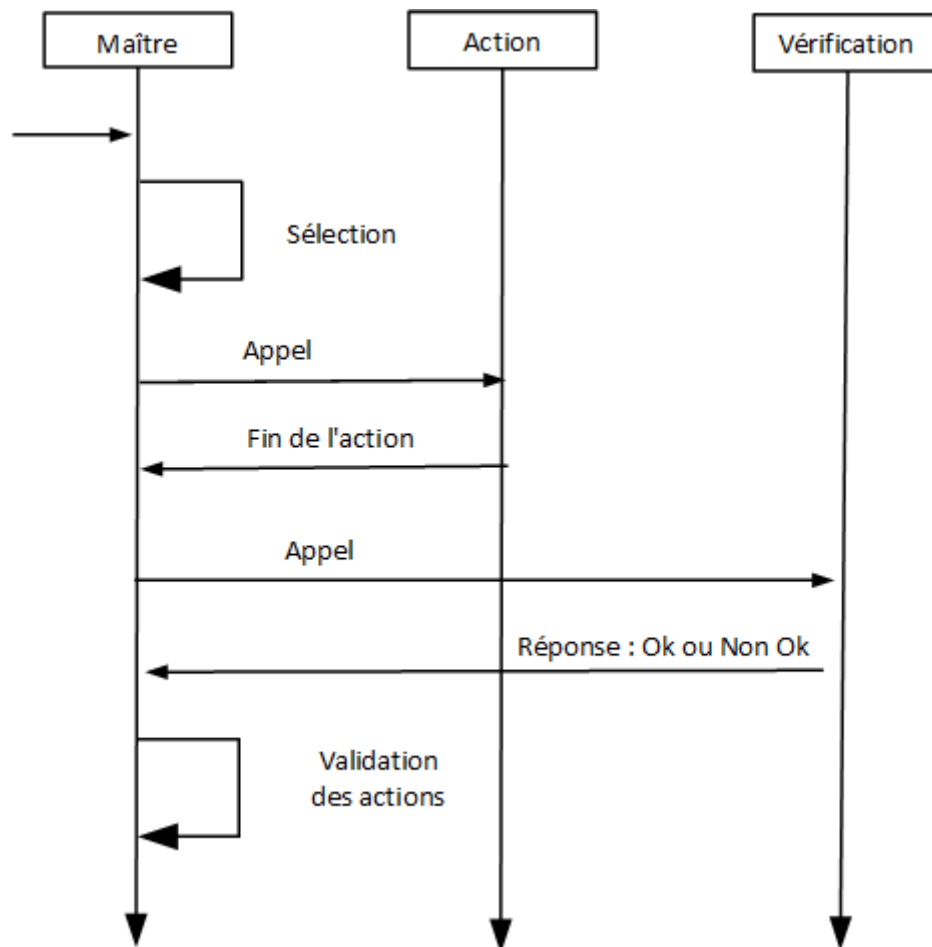


Figure 21: Interaction entre les différents types de modules.

Tout d'abord une sélection des éléments d'action des leviers est réalisée à l'intérieur du champ d'application du levier (dernière colonne du Tableau 11). Puis les modules d'action et de vérification sont appelés.

En fonction du résultat de la vérification, l'action est validée si la réponse est un accord (la nouvelle allocation est enregistrée et elle respecte les contraintes. Par contre le respect de l'objectif n'est pas évalué à ce niveau), sinon le système retourne à son état initial, pour essayer d'appliquer le levier sur d'autres éléments, ou bien appliquer d'autres leviers

Si la première action a été validée, une deuxième action peut être lancée (pour des leviers plus élaborés) La vérification est appelée par le module maître et non pas par l'action : cela permet d'appeler la vérification seule : pour vérifier une allocation complète par exemple (plutôt que de juste vérifier l'action)

III.3.2. Conception technique des modules

À l'aide du formalisme des automates communicants défini précédemment (chapitre 2, IV.3), le comportement de chaque module peut être détaillé. Les appels et les retours entre les modules utilisent le mécanisme des transitions synchronisées $L = (L_e, L_r)$. L'ensemble des leviers utilisés pour la résolution est alors représenté par le réseau d'automates composés de ces différents modules.

Pour chacun des trois types de modules définis précédemment, on peut définir un modèle générique comme présenté ci-dessous (Figure 22, Figure 23, Figure 23).

a. Le modèle générique d'un module maître

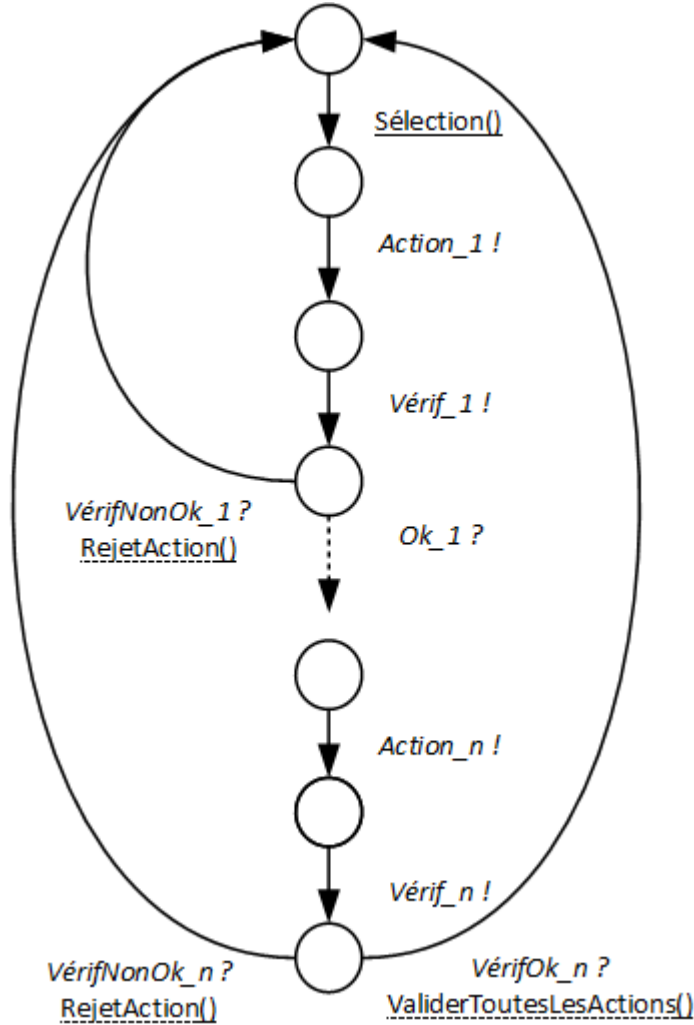


Figure 22: Modèle générique d'un module maître.

Le module maître a pour rôle de coordonner les différentes sous-fonctions réalisant un levier d'action (fonction principale).

Comme illustré par la Figure 22, la première transition $t_{select} \in T$ de ce module permet de sélectionner les éléments sur lesquels le levier va agir (en fonction de la définition du levier et de son champ d'application. Ces valeurs sont enregistrées dans les variables de X par la mise à jour $sélection() \in M$)

Le module maître fait ensuite appel de manière séquentielle à plusieurs modules d'actions élémentaires et de vérification. Ces appels sont séquentiels et ils sont conçus avec un label d'émission $action_i ! \in L_e$ ou $vérif_i ! \in L_e$. La réponse du module de vérification est reçue par le biais des labels de réception $VérifNonOk_i ? \in L_e$ ou $VérifOk_i ? \in L_e$.

Si les contraintes ne sont pas vérifiées, la transition franchie déclenche la mise à jour $\text{RejetAction}() \in M$ et active de nouveau la localité initiale pour que le levier d'action puisse être utilisé une nouvelle fois (nous retrouvons l'état initial, le levier ne pouvant pas être appliqué sur les éléments sélectionnés).

Si les contraintes sont vérifiées, on appelle l'action suivante, sauf si on était dans la dernière localité : dans ce cas, on applique la mise à jour $\text{ValiderToutesLesActions}() \in M$ afin d'enregistrer la nouvelle allocation et la localité initiale est de nouveau activée (Nous retrouvons l'état initial, cependant l'état a été modifié, car des tâches ont été réallouées).

b. Le motif d'un module d'action

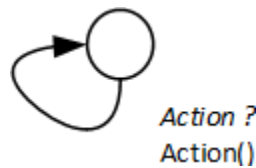


Figure 23: Modèle générique d'un module d'action.

Comme illustré sur la Figure 23, l'état d'initial d'un module d'action permet d'attendre un appel provenant d'un autre module. Cet appel est réalisé lorsque la transition portant le label de réception $\text{action_i} ? \in L_r$ est franchie. Lors du franchissement de celle-ci, la mise à jour $\text{Actions}() \in M$ permet de réaliser l'action et de modifier l'équilibrage. La localité initiale est alors réactivée, le module étant alors prêt pour un nouvel appel.

c. Le motif d'un module de vérification

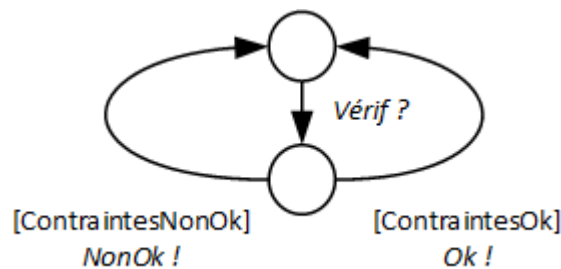


Figure 24: Modèle générique d'un module de vérification.

Comme illustré sur la Figure 24, l'état d'initial d'un module de vérification permet d'attendre un appel provenant d'un autre module. Cet appel est réalisé lorsque la transition portant le label de réception $\text{Vérif_i} ? \in L_r$ est franchie. La localité alors atteinte est la source de deux transitions dont le franchissement renverra le résultat de la vérification grâce aux labels d'émission $\text{VérifNonOk_i} ! \in L_e$ ou $\text{VérifOk_i} ! \in L_e$. Les étiquettes des deux transitions contiennent une garde s'excluant mutuellement $[\text{ContraintesNonOk}] \in G$ et $[\text{ContraintesOk}] \in G$ (à tout instant, une et une seule de ces deux

transitions est franchissable). Cela permet de vérifier que les valeurs des variables de X vérifient les contraintes spécifiées.

III.4. Implémentation des modules avec UPPAAL

III.4.1. Choix du logiciel

Le but de la troisième étape est l'implémentation logicielle de chaque module afin de constituer la librairie prête à être utilisée. Le logiciel choisi doit être capable de réaliser une recherche d'atteignabilité sur l'ensemble des modules modélisant notre système. Dans notre cas il s'agit d'un réseau d'automates communicants. En plus de cela, l'outil doit pouvoir fournir une trace exemple ou contre exemple de la propriété vérifiée qui sera alors la solution cherchée. En effet, une solution est recherchée, et savoir si elle existe ou non n'est pas suffisant.

Le logiciel choisi est le logiciel UPPAAL développé par les universités d'UPPSALA et d'AALBORG (Behrmann et al., 2002). Il a été choisi pour sa simplicité de modélisation des automates génériques grâce à son interface graphique ergonomique et son système de « templates » instanciables. De plus, il est équipé d'un solveur pouvant réaliser la recherche d'atteignabilité et pouvant nous fournir une trace de l'exécution. Celui-ci a aussi l'avantage de pouvoir être appelé par un script, permettant ainsi d'automatiser certaines tâches comme la relance de la résolution avec une nouvelle valeur de l'objectif.¹

De plus, ce logiciel dispose de moyens permettant de réduire l'espace d'états et d'accélérer la recherche. En effet, l'explosion combinatoire apparaît rapidement avec l'augmentation du nombre de tâches ou du nombre de postes de travail. L'exploration de l'espace de solutions est donc long, voire parfois impossible.

La première étape pour réduire ce nombre d'états est de ne modéliser que des états correspondants aux états du système. Pour cela, les outils proposent certaines bonnes pratiques de modélisation (Behrmann et al., 2004) (« A Tutorial on Uppaal ») afin de ne pas alourdir inutilement le modèle :

- Les étapes intermédiaires n'étant que transitoires, elles ne sont pas des états à part entière. UPPAAL propose des localités dites "committed". De par leurs restrictions, celles-ci limitent le nombre d'explosions en évitant le franchissement d'autres transitions (Nb autres transitions évitées multiplié par le nombre de passages par ces localités). Cette suite de localités se comporte alors comme une seule transition.
- De même, certaines variables ne sont que des outils de modélisation et ne représentent pas un état du système. elles peuvent être déclarées meta) (un état = une localité + valeurs des variables, le meta permet de regrouper plusieurs états ne différant que par la valeur de ces variables : les variables d'assignation ne peuvent pas être meta, mais les variables de construction si).

La deuxième action consiste à optimiser le parcours de l'espace d'états. La première amélioration consiste à parcourir l'espace d'états de manière dynamique (à la volée, « on the fly » comme par exemple dans (Hammer et al., 2005)). Afin d'organiser le parcours de l'espace d'états et de ne pas oublier d'évaluer

¹ L'aspect ergonomique a été ici privilégié car notre but est d'étudier la faisabilité de la méthode.

L'optimisation future de celle-ci nécessitera d'étudier les différentes méthodes d'exploration de l'espace d'états et une évaluation de l'impact de l'usage d'un autre outil tel que SPIN ou NuSMV.

certaines états, des algorithmes de recherche ont été proposés par la théorie des graphes. On retrouve ainsi le parcours en largeur (breadth first), en profondeur, (depth first), en profondeur mais avec une part aléatoire (random depth first). Ceux-ci travaillent principalement avec une liste d'états qui n'ont pas encore été explorés.

III.4.2. Implémentation des modules

Chacun des automates défini précédemment est implémenté sous la forme d'un « template » (modèle générique) dans le logiciel UPPAAL. L'instanciation du problème spécifique consistera alors à l'activation des différents modules souhaités et à la définition des valeurs utilisées. Ces paramètres sélectionnés lors de l'instanciation incluent le champ d'application du module et les valeurs des contraintes (le template contient la vérification, mais il faudra adapter les valeurs à chaque problème).

Les règles d'instanciation correspondant à la définition des leviers (comment ils sont réalisés par les modules) ont été définies précédemment. Le système instancié ne pourra donc contenir que certains modules.

Par exemple, pour le levier de déplacement, il faut définir les modules élémentaires suivants : M(déplacer), V(précédence) et A(déplacer) et pour le levier de parallélisation de deux tâches, le modules de haut niveau est composé de 5 modules élémentaires : M(paralléliser), V(parallélisation), V(précédence), A(déplacer_pour_paralléliser) et A(masquer). En effet, pour allouer deux tâches en parallèle, deux actions sont effectuées l'une après l'autre : un déplacement d'une des tâches afin que celles-ci soient allouées sur la même opération puis un « masquage » de la tâche la plus courte car la durée nécessaire pour effectuer deux tâches en parallèle correspond à la durée de la tâche la plus longue.

Une fois chaque automate défini, le réseau est défini comme la composition des n automates représentant les modules : $NA = A_1 \parallel A_2 \parallel \dots \parallel A_n$. Ce modèle général doit ensuite être instancié et activé pour chaque problème spécifique.

IV. Problème spécifique : Instanciation des opérateurs génériques

IV.1. Choix des leviers et de la stratégie de reconfiguration

Afin de résoudre un problème spécifique, la première étape consiste à analyser le problème spécifique afin d'en tirer les données nécessaires et de définir les contraintes et les objectifs. Cette analyse utilise alors l'analyse de la classe de problème. Parmi la librairie de leviers définis précédemment, certains ne pourront pas être appliqués au cas étudié. Ceux qui peuvent être appliqués forment alors un sous-ensemble de leviers possibles noté L .

Si l'analyse révèle un levier n'existant pas dans la librairie, celui-ci peut être ajouté à la librairie (enrichissement de la librairie). Il faudra alors appliquer de nouveau les étapes de conception et d'implémentation relatives à la classe de problème afin de définir comment celui-ci est réalisé, c'est à dire de quels modules élémentaires il est constitué et comment ceux-ci se combinent (III.1 et III.2).

Parmi ces leviers possibles, certains peuvent être privilégiés ou au contraire à éviter. Définir une stratégie consiste à : choisir des leviers et choisir un objectif (en effet, si aucune solution n'existe, on peut préférer élargir la marge de tolérance de l'erreur plutôt que d'activer un levier qui serait difficile à gérer).

Ainsi le choix d'un ensemble de leviers L' extraits de L , définit une stratégie de reconfiguration élémentaire. Contrairement à L qui était imposé par les contraintes, L' est un choix dépendant du contexte et des objectifs. Par exemple, la parallélisation de deux tâches peut être possible sur la ligne considérée, mais le conducteur de ligne souhaite en avoir le moins possible pour limiter les perturbations.

Des règles de priorité doivent donc être définies dans la stratégie globale afin de classer les différentes stratégies élémentaires. Ces différentes stratégies élémentaires doivent donc être ordonnées par un ordre de priorité en fonction des préférences de la ligne.

IV.2. Conception : assemblage des modules techniques

Pour résoudre une stratégie élémentaire, il faut d'abord l'implémenter dans le logiciel. Pour cela, les modules élémentaires correspondant à chaque levier sélectionné doivent être instanciés. Ensuite, avant le lancement de la résolution, l'objectif doit être traduit en propriété d'atteignabilité (comme expliqué dans le chapitre 2, IV.5). Le logiciel peut alors chercher une solution.

La solution trouvée peut alors être appliquée dans l'usine. Si aucune solution est trouvée, il faut alors retenter une autre stratégie de reconfiguration (nouveau choix de levier ou élargissement de la marge de tolérance autour de l'objectif).

Ainsi nous avons défini une heuristique afin de définir les actions à effectuer en fonction des résultats fournis par le logiciel. Cela peut être représenté par un logigramme comme sur la Figure 25. L'heuristique peut aussi être utilisée si l'on souhaite obtenir une solution plus optimale (mais cela nous éloigne du problème réactif traité ici).

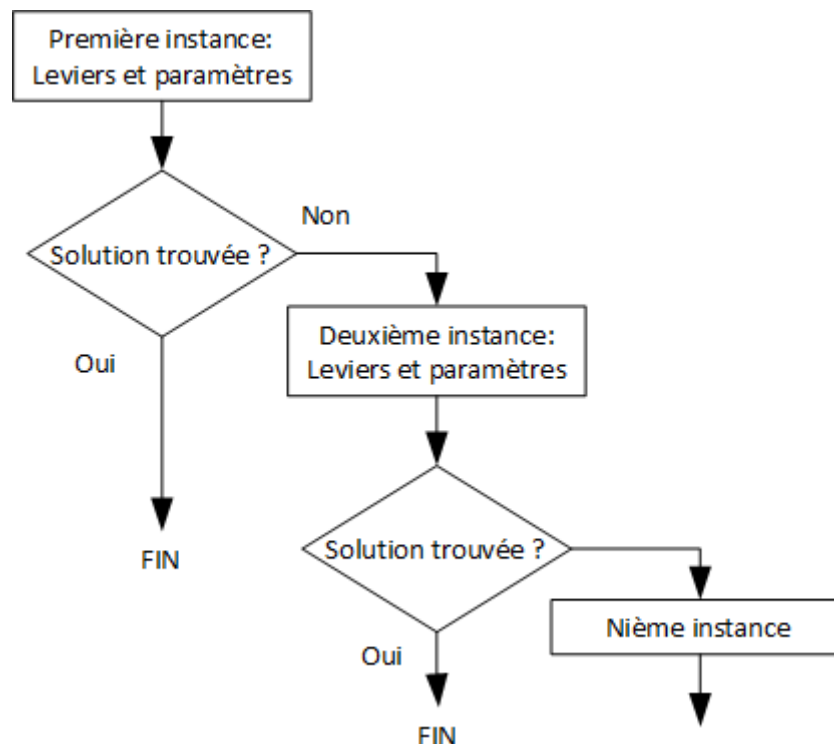


Figure 25: Heuristique de la stratégie de rééquilibrage.

IV.3. Implémentation des modules avec le logiciel UPPAAL

IV.3.1. Implémentation

a. Implémentation des modules

Tout d'abord, l'implémentation consiste à initialiser des différents éléments du problème. Pour cela, les valeurs initiales des variables (de l'ensemble X) du réseau d'automates sont définies. On définit ainsi l'ensemble des tâches T (avec leur durée D_T), l'ensemble des postes de travail W , l'allocation initiale A_w . Lors de l'instanciation, la durée du retard est estimée et celui-ci est pris en compte par la définition des sous-ensembles T_{bis} et W_{bis} .

Ensuite, les modules techniques nécessaires à la réalisation des leviers désirés (et définis précédemment) sont activés. Afin d'affiner cette activation, les champs d'application des leviers sont définis par la définition des ensembles de tâches sur lesquels les leviers peuvent être appliqués et les contraintes restreignant l'action du levier. Ainsi, pour paralléliser deux tâches d'un même poste, l'automate utilisé sera différent de celui utilisé pour paralléliser deux tâches quelconques. En effet, ce deuxième levier nécessite d'abord une action de déplacement des tâches afin de les mettre sur le même poste avant d'appliquer la première action.



Figure 26: Levier « déplacement » : 1 module de sélection, 1 module d'action et 1 module de vérification.

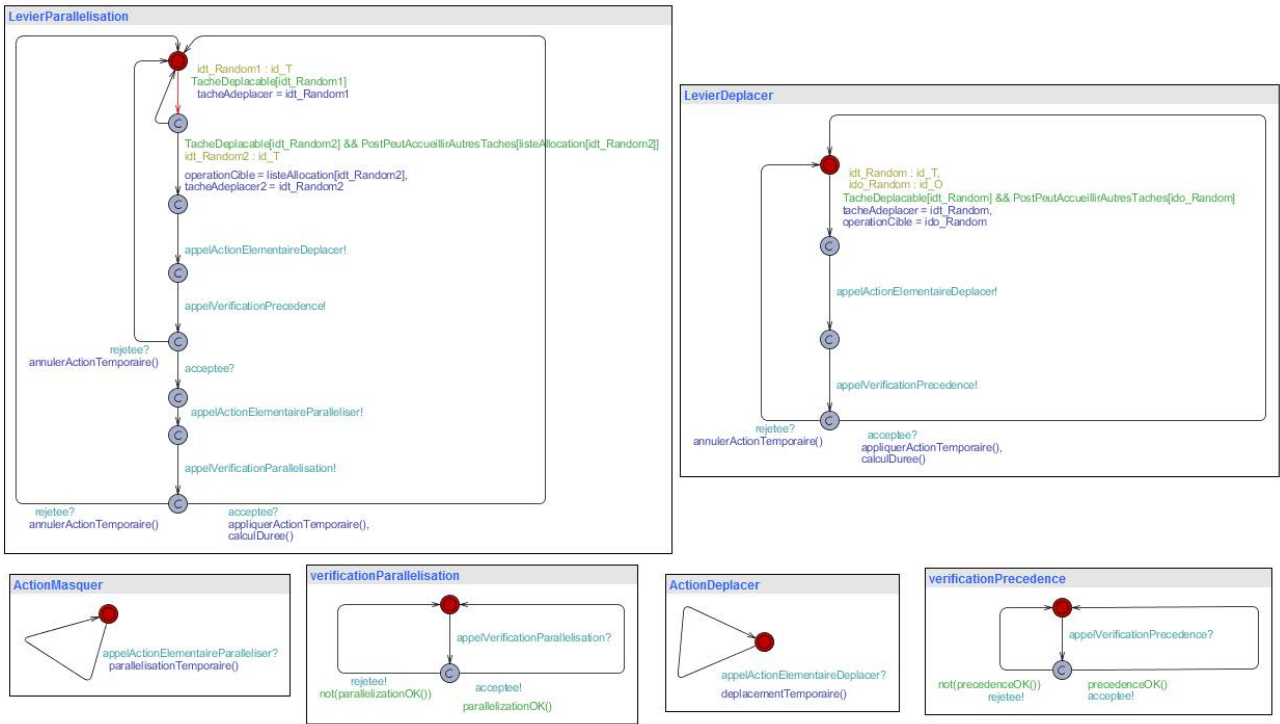


Figure 27: Levier « parallélisation » : 1 module de sélection, 2 modules d'action et 2 modules de vérification.

b. Implémentation de la propriété

Les tâches étant toujours allouées à un poste de travail, nous n'avons pas besoin de vérifier celle-ci. En effet, l'allocation est toujours définie soit par sa valeur initiale, soit par une nouvelle valeur après l'exécution d'un levier d'action. Nous vérifions donc juste la propriété du respect du takt time.

$$(P2) EF(\text{le takt time est respecté}) = EF(\forall w(i) \in W, d_{w(i)} \leq Takt\ Time)$$

avec E le quantificateur d'existence d'un chemin et F le quantificateur temporel de l'éventualité.

La propriété P2 signifie donc : existe-t-il un chemin tel que « le takt time respecté » sera vraie un jour ?

En implémentation, sur UPPAAL EF devient $E\langle \rangle$:

$$(E\langle \rangle (\text{forall } (j:id_O) (\text{sum } (i:id_T) (\text{listeDesDurees}[i] * (j == \text{listeAllocation}[i]))) \leq \text{takt} - \epsilon))$$

c. Implémentation de la stratégie globale

Le rôle du script est de réaliser la stratégie globale de reconfiguration. Nous avons vu qu'elle se traduit par des règles de priorité puis par une heuristique. Pour cela celui-ci doit pouvoir instancier les variables représentant la structure de la ligne d'assemblage comme par exemple la durée des tâches D_i , la liste des postes de travail W ou encore les caractéristiques du retard.

Ensuite, pour chaque instance celui-ci doit activer les modules nécessaires à la réalisation des leviers d'action souhaités en tenant compte des différents champs d'application (voir partie a). De plus, la propriété doit être définie en tenant compte des marges autour du takt time souhaité.

Enfin celui-ci appelle le logiciel de résolution en lui transmettant le modèle instancié et la propriété définie. La réponse du logiciel est alors analysée pour savoir si une solution a été trouvée, et dans ce cas l'afficher, ou bien si aucune solution n'a été trouvée, le script instancie l'étape suivante du logigramme.

```
import os
import sys

model_file = r".\Python\2015_05_22_RTAC_testSOLVER.xml"
properties_file = r".\Python\2015_05_22_RTAC_testSOLVER.q"
trace_file = r".\Python\traceExempleSOLVER.txt"

# Choix des leviers
levier = "deplacer" # "deplacer" OU "paralleliser" OU "deplacer et paralleliser"

# Modifier modèle Choix des modules
if levier == "deplacer":
    modules = "system LevierDeplacer, Deplacer, verificationPrecedence;"
else:
    if levier == "paralleliser":
        modules = "system LevierParallelisation, Masquer, verificationParallelisation, Deplacer,"
    else:
        modules = "system LevierDeplacer, Deplacer, verificationPrecedence, LevierParallelisation"

# conversion des données
sys.argv=["ModifyDataFormat.py", Nt, DurationList, PrecedenceConstraintsNumber, PrecedenceConstra]
execfile("ModifyDataFormat.py")

# instanciation du modèle
sys.argv=["ModifyModel.py", model_file, takt, No, Nt, DurationListTXT, PrecedenceConstraintsNumbe]
execfile("ModifyModel.py")

# instanciation de la propriété
sys.argv=["ModifyProperties.py", properties_file]
execfile("ModifyProperties.py")

# Appel du solveur
os.system(".\SolverLaunch.bat " + model_file + " " + properties_file + " " + trace_file)

# Analyser réponses solveur
execfile("AnaylserRetourSolveur.py")
```

Figure 28: Résolution d'une instance par un script python.

Le lancement d'une nouvelle résolution (par le script ou manuellement) peut permettre d'obtenir une deuxième solution différente de la première (des solutions acceptables, pas forcément la solution optimale), laissant ainsi le choix de laquelle appliquer en fonction des éléments de décision non pris en compte par le modèle. (D'autres éléments d'implémentation sont précisés dans l'annexe 2.)

V. Conclusion

Dans ce chapitre, nous avons présenté la méthode de génération de modèles qui est l'apport principal de la thèse. Celle-ci a pour but de concevoir un outil de résolution de plusieurs rééquilibrages dynamiques. Pour cela nous avons vu dans le chapitre 1 que la méthode de résolution devait être rapide, mais aussi générique.

C'est pourquoi la méthode proposée ici se base sur une analyse fonctionnelle de la classe de problème afin de concevoir une librairie de « modules standards » permettant ainsi une adaptation rapide à chaque cas spécifique que l'on souhaite résoudre. Ainsi, lorsque seul le levier de « déplacement » est activé, le modèle obtenu est différent du modèle SED décrit à la fin du chapitre 2. Néanmoins, les deux espace d'états sont similaires. La différence entre les deux approches de modélisation trouve alors son sens lorsqu'un autre levier d'action doit être activé pour résoudre un autre problème spécifique.

L'application est ensuite aisée grâce à un ensemble de règles d'application illustrées par des exemples.

Dans le chapitre suivant, nous allons présenter différents cas d'étude afin d'évaluer si la généricité est satisfaisante.

Chapitre 4 : Application de la proposition

Table des matières

I. Introduction.....	89
II. Présentation générale des cas d'étude.....	89
II.1. L'entreprise.....	89
II.2. Les cas d'étude.....	90
III. Premier cas d'étude.....	92
III.1. Description.....	92
III.2. Application de la méthode.....	92
IV. Deuxième cas d'étude.....	94
IV.1. Description.....	94
IV.2. Application de la méthode.....	94
IV.2.1. Phase d'analyse.....	94
IV.2.2. Phase de conception.....	95
IV.2.3. Phase d'implémentation.....	96
IV.3. Résolution.....	96
V. Validation de la méthode proposée par une analyse du troisième cas d'étude.....	97
V.1. Description du cas d'étude.....	97
V.2. Description des scénarios et des facteurs étudiés.....	98
V.3. L'analyse.....	99
V.3.1. Analyse des effets des facteurs sur le temps de résolution.....	100
V.3.2. Analyse des effets des facteurs sur le nombre de changements.....	102
VI. Conclusion.....	103

I. Introduction

Le but de ce chapitre est d'appliquer, mais aussi de valider la méthode proposée. Pour cela nous la mettrons en œuvre sur différents problèmes de rééquilibrage dynamique de l'entreprise Trane afin d'évaluer l'adaptabilité de la méthode d'initialisation du modèle lorsque la structure de la ligne d'assemblage varie (nombre de postes, de tâches, de contraintes, etc.), mais aussi lorsque divers leviers d'action sont activés.

Ces applications sont ensuite enrichies par une analyse structurée d'un cas d'étude composé d'une seule ligne d'assemblage mais de nombreux scénarios. Nous validerons alors la méthode sur plusieurs cas, enfin cette analyse nous permettra aussi de mettre en évidence plusieurs perspectives intéressantes.

II. Présentation générale des cas d'étude

II.1. L'entreprise

Afin de satisfaire ses nombreux clients, qu'ils soient des particuliers ou d'autres entreprises, l'entreprise Trane propose une large gamme de produits de climatisation ou de traitement de l'air avec de nombreuses variantes (Figure 29). La spécificité de chaque produit et le volume de vente font que l'entreprise se situe dans une classe de typologie de systèmes de production connue sous le nom de « personnalisation de masse » (mass customization).

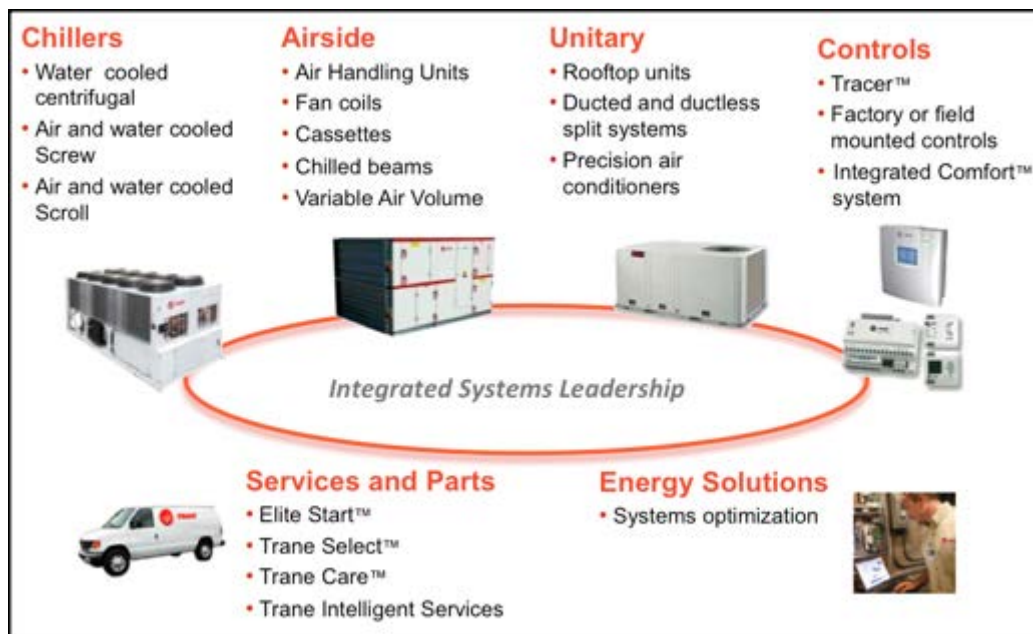


Figure 29: Aperçu des produits et des services proposés par l'entreprise Trane.

Cette personnalisation de masse a poussé l'entreprise, pour la conception de ces lignes, à adopter les principes issus de la DFT permettant ainsi d'uniformiser ses méthodes de production sur l'ensemble de ses 27 sites de production à travers le monde. Ses lignes sont donc cadencées suivant des takt times. L'entreprise souhaite alors mettre en place un système de rééquilibrage dynamique pour réagir aux différents aléas du terrain. Dans cette partie, nous nous concentrerons sur la recherche de solution de rééquilibrage. La détection du retard et l'application de la solution sont supposées réalisables : le laps de

temps entre la détection du défaut et la réaction est supposé court et la diffusion des consignes aux opérateurs est supposé rapide. Divers travaux en collaboration entre l'entreprise et le CRAN (El-Haouzi, 2008 ; Giovannini, 2015) ont permis de mettre en place le concept de « système contrôlé par le produit ». L'application développée (annexe 3) permet, entre autre, en fonction des codes à barres identifiant les produits d'afficher la nomenclature, les tâches et les contrôles à réaliser, d'enregistrer les temps de passage de chaque produit sur chaque poste. De plus, l'usage d'un système de type « kitting² » rend ce rééquilibrage facilement applicable. Néanmoins, d'autres cas doivent être pris en compte comme par exemple le fait que certaines tâches ne peuvent être effectuées que sur les postes disposant de l'outillage adapté et des opérateurs qualifiés (comme par exemple la soudure). Ces cas particuliers sont alors pris en compte dans la méthode de résolution sous la forme de contraintes.

Pour valider notre méthode, plusieurs cas vont maintenant être étudiés. En effet, la variété des usines et des lignes de l'entreprise nécessitent de vérifier que la méthode peut s'adapter.

II.2. Les cas d'étude

Le premier cas d'étude, basé sur le scénario 1 présenté précédemment est utilisé ici pour valider la méthode de modélisation et de résolution lorsque le seul levier d'action activé est le déplacement des tâches d'un poste à un autre. En effet, la ligne considérée est une ligne où le reséquencement n'est pas accepté et où le travail en parallèle n'est pas possible car les postes ne peuvent pas accueillir plusieurs opérateurs.

Cette ligne, basée en république tchèque est spécialisée dans l'assemblage de « fancoils ». Ces petites unités sont destinées à être placées sur des bouches de ventilation afin de réchauffer ou de refroidir l'air qui en sort grâce à un échangeur thermique entre l'air et un fluide frigorigère fourni par une autre machine de production de froid. Comme ces produits sont destinés à être répartis dans l'ensemble d'un bâtiment, ceux-ci sont assemblés en grand nombre et leurs puissances frigorifiques respectives sont faibles donc leurs dimensions aussi.



Figure 30: Un « fancoil » Trane.

Afin d'étudier l'adaptabilité de la méthode lorsque la structure de la ligne est modifiée, le deuxième cas d'étude est alors une ligne différente où plusieurs leviers peuvent être appliqués. Cependant certains étant plus pénalisant en termes de coût que d'autres, ceux-ci ne sont appliqués qu'en cas de nécessité. Une stratégie globale de reconfiguration est donc définie et appliquée afin de chercher les solutions utilisant les

² Le kitting consiste à utiliser un chariot contenant les pièces à assembler pour un ensemble de postes et suivant le produit à assembler, la pièce nécessaire est ainsi toujours disponible au moment où on en a besoin.

leviers d'action les moins pénalisants. Cela constitue le deuxième besoin d'adaptabilité de la méthode. Ces deux besoins d'adaptabilité de la méthode (variations structurelles et fonctionnelles) sont illustrés sur la Figure 31 ci-dessous.

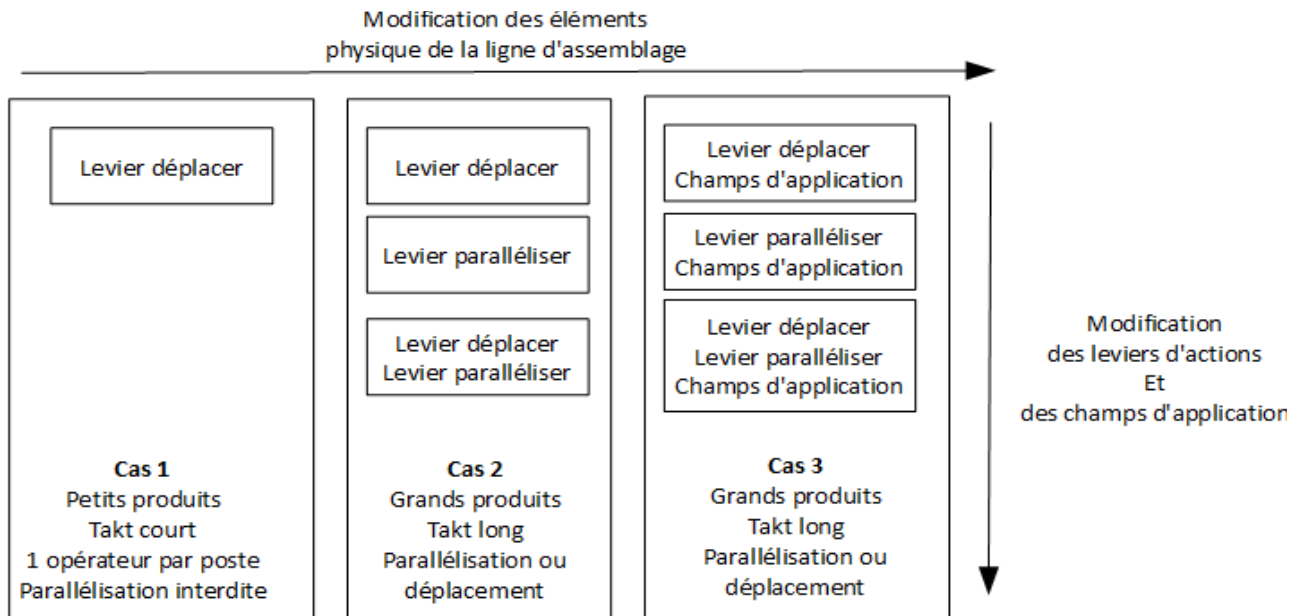


Figure 31: Les différences entre les cas d'étude étudiés.

Le troisième cas d'étude a pour objectif d'élargir la validation à plusieurs situations mettant en évidence l'impact de différents facteurs comme le choix des champs d'action en plus de la spécification des leviers. Ceux-ci permettent en effet de limiter le nombre de déplacements de tâches pour que la réallocation ne perturbe pas trop la production. Nous verrons aussi qu'ils peuvent être utilisés pour réduire l'espace d'états afin d'éviter l'explosion combinatoire et de trouver une solution pour des problèmes de plus grandes dimensions. Pour cela, différents scénarios se basant sur la même ligne d'assemblage seront étudiés selon un protocole expérimental rigoureux.

Cette ligne d'assemblage correspond à une des lignes de l'usine, en France, dans les Vosges (88). Les produits qui y sont assemblés sont de la famille des « air cooled » (voir les « chillers » sur la Figure 29). Ceux-ci étant dédiés à l'installation sur le toit de bâtiments de différentes, leurs puissances frigorifiques couvrent une large gamme. Cela engendre donc une grande diversité des produits, mais aussi de grandes dimensions pour certains. Les tâches effectuées sur cette ligne sont principalement des tâches d'assemblage des sous-ensembles sur le châssis du produit. Les contraintes de précédence sont alors peu nombreuses et plusieurs opérateurs peuvent travailler simultanément sur le même produit. À cela s'ajoute des tâches de soudure permettant de fermer le circuit du fluide frigorifique qui doivent être réalisées sur les postes disposant de l'équipement et dans des conditions de sécurité impliquant qu'un seul opérateur travaille sur le produit. Ces tâches ne pourront donc pas être parallélisées. Dans ce cas d'étude, nous ne considérons pas les tâches de test. En effet, celles-ci doivent être réalisées à des postes précis et avec des produits dont l'état d'avancement est défini. La ligne est alors découpée en plusieurs portions distinctes où le rééquilibrage peut être appliqué sur chacune. (La liste des durées des tâches et de leurs allocations initiales est fournie dans l'annexe 4.)

III. Premier cas d'étude

III.1. Description

Ce premier cas d'étude ayant été présenté dans le chapitre 2 (scénario 1), nous pouvons valider la méthode de modélisation et de résolution lorsque le seul levier d'action activé est le déplacement des tâches d'un poste à un autre.

Nous pourrions alors comparer les résultats obtenus avec ceux obtenus par la méthode ILS et par la recherche d'atteignabilité sur un réseau d'automates communicants modélisant les tâches et les postes de travail de la ligne.

Le problème est défini ainsi : $W = \{W_1, W_2, W_3\}$ et $T = \{t_i, i \leq 19\}$ avec un takt de 266 secondes.

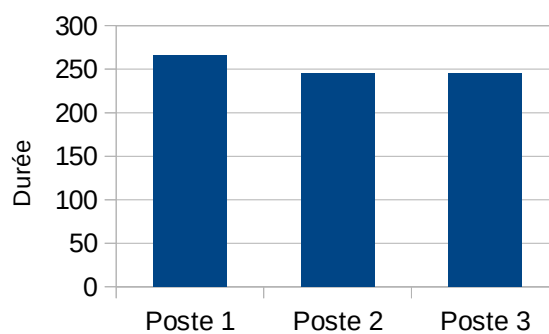


Figure 32: Équilibrage initial du premier cas d'étude.

Le retard considéré est appliqué sur la tâche t_{4r} au poste W_1 et d'une durée de 40 secondes. On a alors $W_{bis} = \{W_2, W_3\}$ et $\text{card}(T_{bis}) = 16$.

III.2. Application de la méthode

La première étape consiste à réaliser l'analyse du problème spécifique afin d'en identifier les leviers d'actions. Dans ce cas, une seule stratégie élémentaire de reconfiguration est utilisée : le levier déplaçant une tâche d'un poste de travail vers un autre. Le champ d'application défini autorise que l'ensemble des tâches déplaçables (T_{bis}) puisse être sur l'ensemble des postes de travail disponibles (W_{bis}). Le levier d'action de déplacement est alors activé. Celui-ci est composé des modules : maître(déplacer), vérification(précédence) et action(déplacer) (Figure 33).



Figure 33: Modèle du levier de déplacement.

Dans ce réseau d'automates, l'état initial est composé de 3 localités actives et est défini par la valeur de l'allocation initiale. D'après les règles de franchissement des transitions, initialement, seul le premier automate peut évoluer. La transition correspondant à une sélection de la tâche à déplacer et du poste sur lequel on souhaite la déplacer, le nombre d'évolutions possibles est le produit $\text{Card}(T_{\text{bis}}) * \text{Card}(W_{\text{bis}})$. Le nombre d'automates est donc moins élevé que dans le modèle où chaque automate représente un poste de travail ou une tâche, mais le nombre d'états possibles est le même.

De même que précédemment, les indicateurs d'évaluation des résolutions sont : le nombre de changements pour passer de l'équilibrage initial à la solution de réallocation, la durée mise pour trouver cette solution et le temps de cycle de la nouvelle allocation :

	Nb de changements	Temps de résolution en s.	Temps de cycle en s
Test 1	3	0,015	285
Test 2	11	0,219	287
Test 3	3	0,046	291
Test 4	2	0,031	280
Test 5	11	3,822	275
Test 6	6	36,473	284
Test 7	2	0,109	276
Test 8	1	0,031	279
Test 9	2	0,016	280
Test 10	12	0,046	282
Moyenne	5,4	4,0808	281,9

Tableau 12: Résultats de la recherche d'atteignabilité pour le premier cas d'étude.

Nous remarquons alors que les performances sont similaires au modèle précédent. Cette méthode remplit donc le premier besoin qui est celui de la résolution du problème de rééquilibrage dynamique. Cependant, contrairement aux modèles testés dans le chapitre 2, d'autres leviers d'action ont déjà été définis par l'analyse de la classe de problème. Nous allons donc appliquer cette méthode à une autre ligne sur laquelle différents leviers peuvent être appliqués.

IV. Deuxième cas d'étude

IV.1. Description

Le deuxième cas d'étude se base sur la ligne 3 définie dans le chapitre 2 pour l'étude de plusieurs scénarios. Nous nous intéressons ici au scénario pour lequel aucune solution n'était trouvée avec le levier de déplacement. Ainsi, une autre résolution doit être lancée avec de nouveaux leviers. Cela permettra de valider l'adaptabilité de la méthode de modélisation.

Le problème est défini ainsi : $= \{W_o, o \leq 8\}$ et $T = \{t_i, i \leq 310\}$ avec un takt de 4462 secondes.

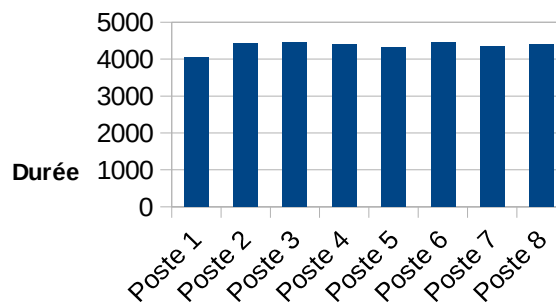


Figure 34: Équilibrage initial du deuxième cas d'étude.

Le retard considéré est appliqué sur la tâche t_{150} au poste W_5 et d'une durée de 100 secondes. On a alors $W_{bis} = \{W_5, W_6, W_7, W_8\}$ et $\text{card}(T_{bis}) = 159$.

IV.2. Application de la méthode

IV.2.1. Phase d'analyse

Lorsque le retard est détecté sur la ligne (sur la tâche t_{150} dans ce cas d'étude), la première étape de la méthode est de choisir les leviers d'actions qui doivent être privilégiés.

L'entreprise souhaite rééquilibrer la ligne pour absorber les perturbations dues à l'aléa, tout en minimisant l'impact des reconfigurations sur la ligne d'assemblage. En effet, chaque levier a un coût d'application car celui-ci engendre des perturbations (par exemple le risque d'erreur peut augmenter ou de la main d'œuvre supplémentaire peut être nécessaire). Ces perturbations doivent être inférieures aux perturbations qu'engendreraient le retard s'il n'était pas corrigé.

Ainsi, dans ce cas d'étude, l'entreprise préfère déplacer des tâches d'un poste à un autre plutôt que de paralléliser. En effet, la parallélisation nécessite le déplacement d'un opérateur alors que le déplacement de la tâche ne nécessite que le déplacement des consignes de travail, la matière première accompagnant le produit dans un kit. De plus, le levier d'action devra d'abord, si possible, s'appliquer aux postes voisins afin

de maintenir les perturbations de la manière la plus locale possible, toujours dans un but de minimiser l'impact d'un rééquilibrage.

IV.2.2. Phase de conception

Une fois les leviers définis et leurs règles de priorités établies par l'analyse, la deuxième étape de la méthode consiste à concevoir les modules et l'heuristique de reconfiguration globale.

Dans notre cas, le levier qui a la plus grande priorité est le levier de déplacement appliqué sur un nombre minimal de postes. Cependant, si aucune solution n'est possible, l'espace de travail est élargi en permettant les déplacements de tâches sur l'ensemble de la ligne. Enfin, si cela ne suffit pas, elle accepte de déplacer des opérateurs d'un poste à un autre afin qu'ils viennent travailler en parallèle d'un autre opérateur : c'est la parallélisation des tâches. La dernière étape consiste à rechercher avec les 2 leviers activés avec un domaine d'application couvrant l'ensemble de la ligne d'assemblage. Cette stratégie de résolution peut alors être représentée par le logigramme suivant (Figure 35).

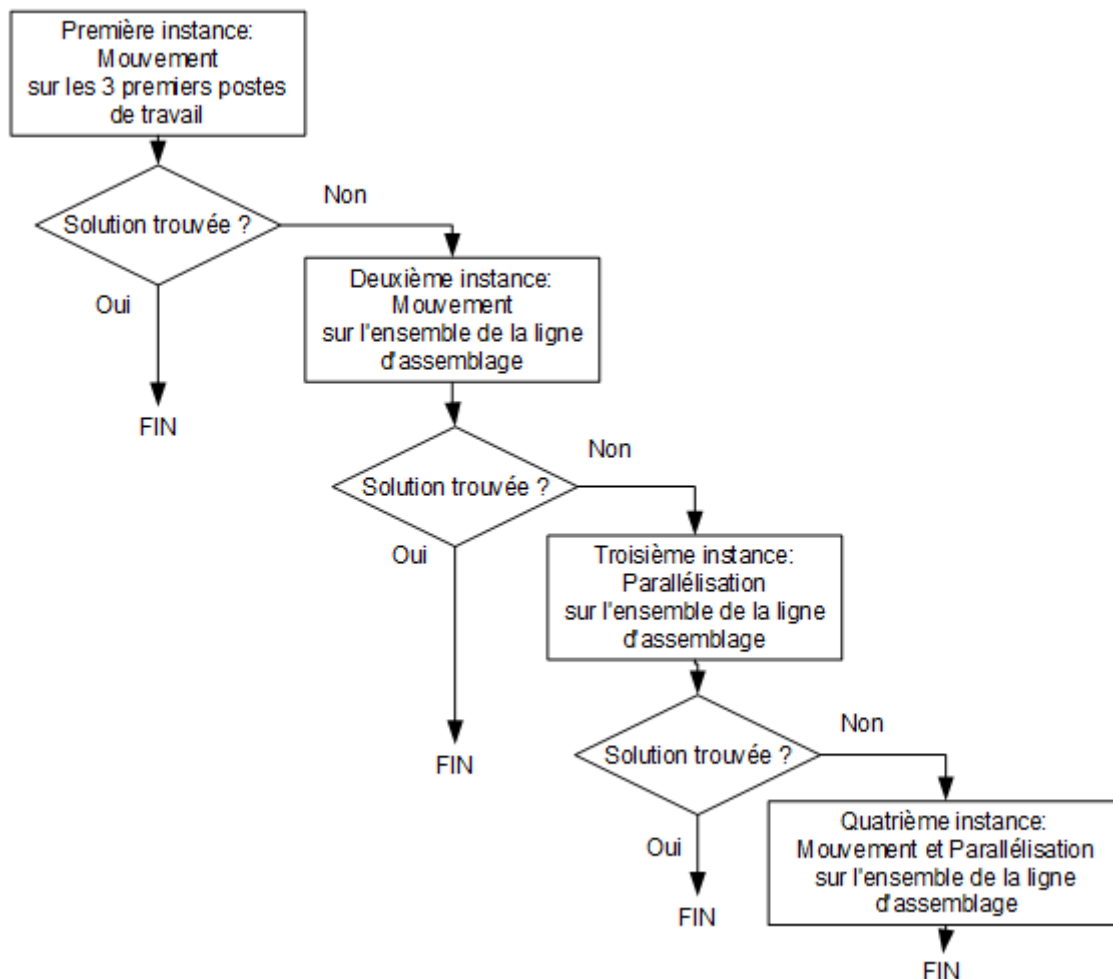


Figure 35: Stratégie de reconfiguration du deuxième cas d'étude.

Pour chaque étape de l'heuristique, un ensemble de leviers d'action est utilisé, il faut donc définir à chaque fois les modules nécessaires. Pour cela les règles fixées lors de l'analyse de la classe de problèmes sont utilisées. Ainsi, pour utiliser le levier de parallélisation, il faut activer cinq modules :

maître(paralléliser), vérification(parallélisation), vérification(précédence), action(masquer) et action(déplacer)

IV.2.3. Phase d'implémentation

Chaque module étant implémentés sous UPPAAL, le lancement de la résolution consiste à initialiser les variables (durées de tâches, allocations initiales, etc.) et à activer les modules nécessaires pour résoudre le cas proposé par l'étape de l'heuristique.

Si, seul le levier de déplacement est activé, 3 modules sont instanciés. Pour l'activation du levier de parallélisation, il faut activer 5 modules comme décrit précédemment. Et si les deux leviers sont pris en compte, 6 modules sont utilisés (certains étant communs aux deux leviers) comme le montre la Figure 36.

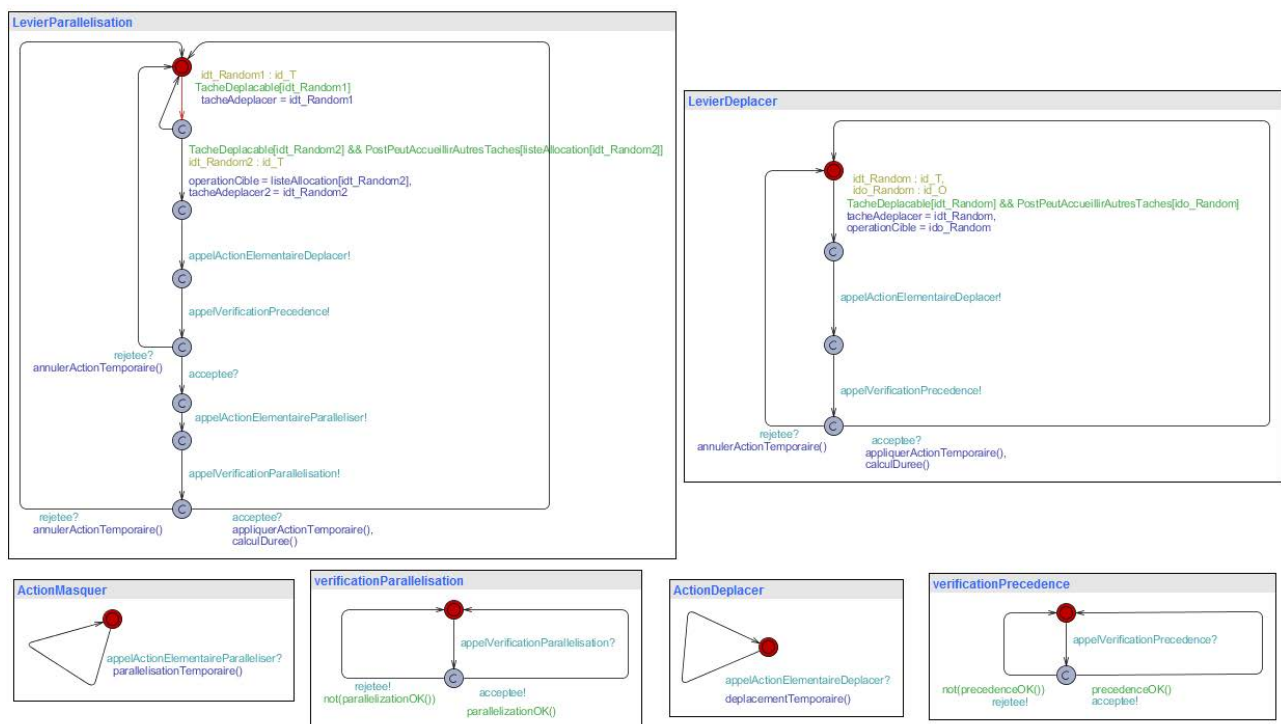


Figure 36: Exemple de résolution du deuxième cas.

Une fois ces leviers définis, la vérification de la propriété est lancée. Il faut ensuite analyser les résultats. En effet, si une solution est fournie, elle doit être appliquée dans l'usine, sinon, on doit tenter une nouvelle résolution avec de nouveaux paramètres comme défini par l'heuristique de reconfiguration globale.

Un script Python a été développé dans le but d'automatiser cette tâche en créant le modèle, en activant les leviers choisis, de définir la marge d'erreur de la propriété à vérifier, de lancer la vérification et de prendre une décision en fonction du résultat retourné.

IV.3. Résolution

De même que précédemment, les indicateurs d'évaluation des résolutions sont : le nombre de changements pour passer de l'équilibrage initial à la solution de réallocation, la durée mise pour trouver cette solution et le temps de cycle de la nouvelle allocation. Le nombre de changements est, cette fois-ci, détaillé afin de différencier le nombre de parallélisations et le nombre de déplacements.

Dans ce cas d'étude, la première résolution n'aboutit pas. On passe donc à la deuxième étape de l'heuristique, qui n'aboutit toujours pas. En effet, il s'avère que la marge disponible sur l'ensemble des postes restants n'est pas suffisante pour absorber le retard. Aucune solution n'existe et le logiciel peut soit parcourir l'ensemble de l'espace de solutions et nous dire que la propriété n'est pas vérifiée, soit subir l'explosion combinatoire et saturer sa mémoire avant de trouver une solution. La parallélisation semble alors la seule voie possible.

La troisième résolution est donc lancée en activant uniquement la parallélisation de deux tâches se trouvant sur le même poste (afin de limiter l'impact sur la ligne). Cette explosion combinatoire peut être réduite en adaptant certains paramètres comme restreindre le champ d'action des leviers d'action ou utiliser d'autres méthodes de parcours etc. Cette étude est l'un des objectifs du troisième cas d'étude.

V. Validation de la méthode proposée par une analyse du troisième cas d'étude

V.1. Description du cas d'étude

L'objectif de ce troisième cas d'étude est de tester l'applicabilité de la méthode à une instance de plus grandes dimensions et de valider notre approche sur plusieurs scénarios. Afin de structurer cette analyse, un protocole expérimental s'appuyant sur un plan d'expériences a été établi permettant ainsi de tester de manière structurée et exhaustive différents scénarios et plusieurs façons d'y faire face. Par exemple, nous prendrons en compte le choix des champs d'application de chaque levier lors de leur activation. En effet ceux-ci peuvent être à la fois des contraintes de l'entreprise (déplacement au poste $i+1$) et un moyen de guider la recherche de la solution.

Le cas d'étude s'appuie donc sur une ligne d'assemblage composée de 8 postes et de 309 tâches : $W = \{W_o, o \leq 8\}$ et $T = \{t_i, i \leq 310\}$. Cette ligne est déjà celle considérée dans le deuxième cas d'étude, cependant, dans cette analyse, le retard pourra survenir plus tôt sur la ligne, augmentant ainsi le nombre de postes concernés par la reconfiguration. Nous pouvons visualiser l'équilibrage initial de cette ligne ainsi que son takt de 4462 s (en rouge) et une marge epsilon de 100 s (en violet) sur le graphique ci-dessous (Figure 37).

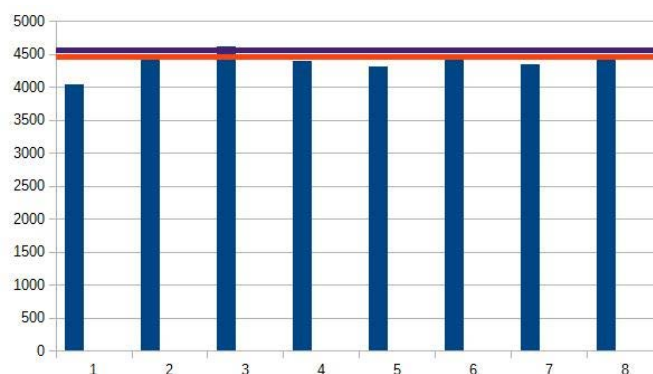


Figure 37: Équilibrage initial du troisième cas d'étude pour l'analyse.

V.2. Description des scénarios et des facteurs étudiés

Cet équilibre initial est alors appliqué en production. Celui-ci peut alors être perturbé par la survenue d'un aléa pouvant apparaître sur n'importe quel poste de la ligne. Afin d'étudier plusieurs cas, nous considérons en premier facteur, l'exemple d'un retard apparaissant sur un poste de début de ligne (sur la tâche t_{64}), un au milieu de ligne (t_{150}) et enfin un, détecté sur un des postes de la fin de la ligne (t_{250}). Ces trois cas d'apparition du retard sont présentés dans le Tableau 13.

	Cas 1	Cas 2	Cas 3
Durée du retard	150s	150s	150s
Tâche en retard	t_{64}	t_{150}	t_{250}
Poste impacté par le retard	W_3	W_5	W_7
W_{bis}	$\{W_3, W_4, W_5, W_6, W_7, W_8\}$	$\{W_5, W_6, W_7, W_8\}$	$\{W_7, W_8\}$
$card(T_{bis})$	245	159	60
Marge totale des postes disponibles	261	192	35

Tableau 13: Données des cas de l'analyse pour le plan d'expériences.

Lorsqu'un aléa est détecté, la durée du retard engendré est estimée. L'heuristique globale de reconfiguration s'assure alors d'activer la résolution. Ainsi, plusieurs méthodes de résolution peuvent être appliquées en faisant varier les leviers d'actions activés (facteur 2) et leurs champs d'application (facteur 3, qui correspond soit à l'ensemble de la ligne soit à la paire constituée du poste le plus en retard et du poste le moins chargé). À cela nous pouvons ajouter le choix de la méthode de parcours de l'espace d'états (facteur 4) : la recherche en largeur (breadth first) ou la recherche en profondeur (depth first) avec une composante aléatoire (random).

Les facteurs de l'analyse, ainsi que leurs niveaux sont présentés dans le Tableau 14.

Facteurs	Niveaux
Position du retard (3 modalités)	Cas 1, W_3
	Cas 2, W_5
	Cas 3, W_7
Lever d'action (3 modalités)	Déplacer
	Paralléliser
	Déplacer et Paralléliser
Champs d'application des leviers (2 modalités)	Restreint à deux postes
	Ligne complète
Méthode de parcours de l'espace d'états (2 modalités)	En largeur
	En profondeur aléatoire

Tableau 14: Description des facteurs étudiés.

Ce plan d'expériences a permis de valider la méthode en testant ses résultats en fonction des combinaisons des différents facteurs. Pour chaque expérience nous avons mesuré le temps de résolution ainsi que le nombre de changements à effectuer pour appliquer la solution. Les résultats sont présentés sur l'annexe 5.

V.3. L'analyse

Les facteurs précédents peuvent être combinés de 36 façons différentes ($3 \times 3 \times 2 \times 2$). Chaque combinaison est testée 5 fois (répétitions) pour estimer l'existence de facteurs éventuels de bruit. Ce nombre de réplifications porte ainsi le nombre d'expériences réalisées à 180 tests. Tous ces essais ayant aboutis à une solution en moins d'une minute, cela nous conforte dans le choix de la méthode et de sa validité. Notons cependant qu'un cas n'a pas pu aboutir à une solution dans ce temps limite.

En plus de montrer la validité de notre méthode sur plusieurs cas différents, cet ensemble de tests nous a aussi permis d'analyser l'effet des différents facteurs principaux (Figure 38) et de leurs interactions (Figure 39) sur les objectifs considérés. L'analyse des résultats a aussi permis de répondre à certaines questions : quels sont les facteurs les plus influents ? existent-ils des facteurs qui n'ont pas été pris en compte ? quelle est la méthode de résolution à appliquer en fonction de chacun des cas ?

Par conséquent, diverses perspectives portant sur les éléments à inclure pour enrichir la méthode proposée s'offrent à nous.

V.3.1. Analyse des effets des facteurs sur le temps de résolution

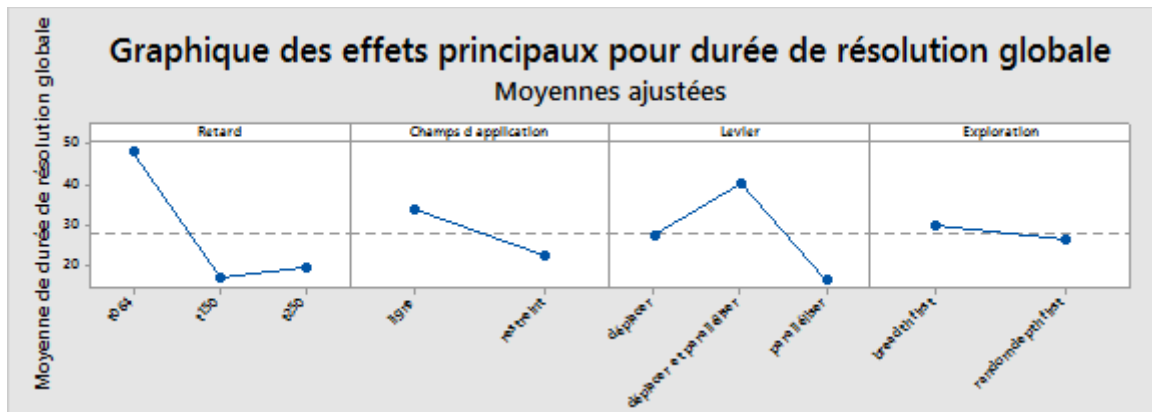


Figure 38: Représentation graphique des effets principaux sur la durée de résolution.

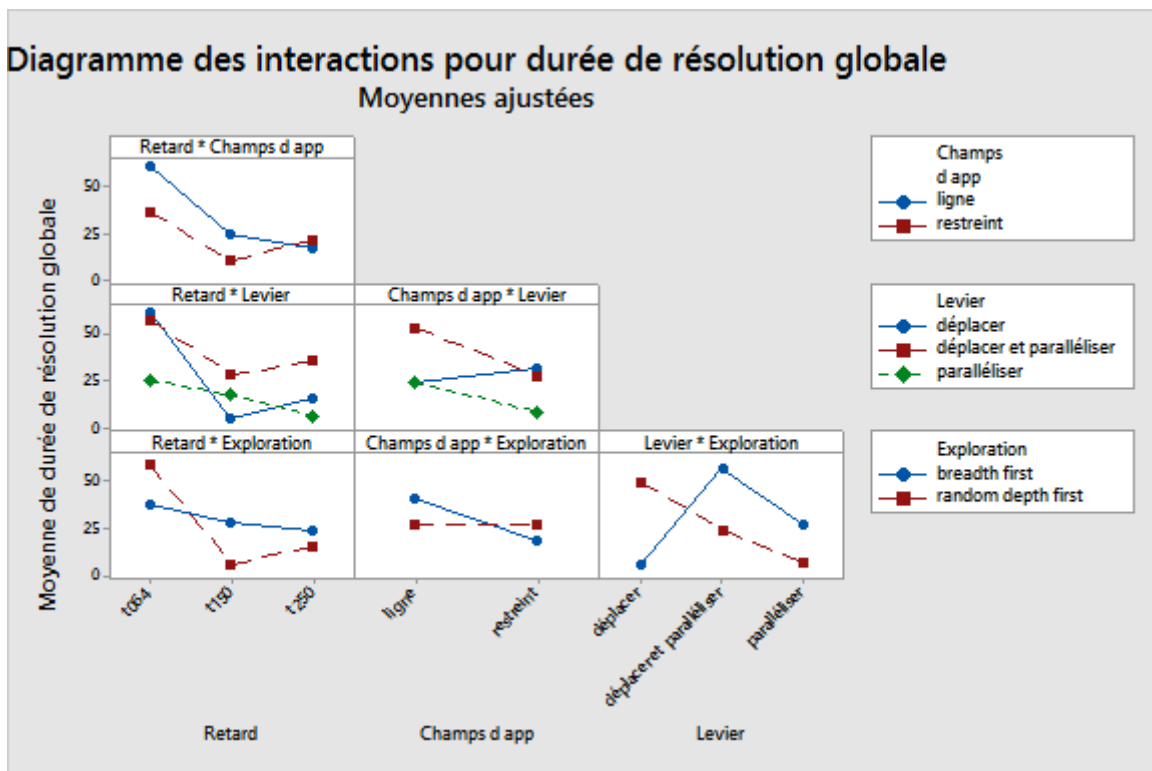


Figure 39: Diagramme des interactions pour sur la durée de résolution.

L'analyse de l'effet d'un facteur passe par le calcul de l'écart entre le résultat lié au niveau du facteur dans une des modalités et la moyenne générale (qui est ici de 29,5 s comme illustré par le trait en pointillés sur le graphique des effets principaux (Figure 38)). Par exemple, l'effet du facteur « position du retard » au niveau de la tâche t_{64} (c-à-d au début de la ligne) « ajoute » 22,5 secondes à la moyenne. Nous pouvons observer sur cette figure que le facteur « position du retard » est fortement significatif. Le « champs d'application », quand il est au niveau nommé « restreint » a un effet positif sur l'objectif de minimisation du temps car il réduit le temps de résolution de 4,5 secondes. L'inverse se produit quand le niveau « ligne » de ce même facteur est choisi. L'effet du facteur « exploration » peut être considéré comme négligeable (pas significatif). Ces premières observations graphiques sont détaillées par l'analyse statistique de la

variance (Figure 40) montrant ainsi que la position du retard, les leviers d'action et les champs d'application sont des facteurs significatifs (resp $p = 0$, $p = 0$ et $p = 0,002$) car p étant inférieur à 0,01, il y a une très forte présomption contre l'hypothèse nulle. Concernant le facteur « exploration » ($p = 0,802$), p est supérieur à 0,1 donc il n'y a pas de présomption contre l'hypothèse nulle et donc ce facteur n'est pas significatif.

Analyse de variance					
Source	SomCar			Valeur	
	DL	ajust	CM ajust	Valeur F	de p
Modèle	35	286939	8198,2	22,66	0,000
Linéaire	6	67030	11171,7	30,88	0,000
Retard	2	45913	22956,3	63,45	0,000
Champs d application	1	3556	3555,9	9,83	0,002
Levier	2	17539	8769,3	24,24	0,000
Exploration	1	23	22,9	0,06	0,802
Interactions à 2 facteur(s)	13	131967	10151,3	28,06	0,000
Retard*Champs d application	2	3905	1952,3	5,40	0,005
Retard*Levier	4	20533	5133,2	14,19	0,000
Retard*Exploration	2	21941	10970,4	30,32	0,000
Champs d application*Levier	2	13720	6860,2	18,96	0,000
Champs d application*Exploration	1	9075	9074,8	25,08	0,000
Levier*Exploration	2	62794	31396,8	86,78	0,000
Interactions à 3 facteur(s)	12	79378	6614,8	18,28	0,000
Retard*Champs d application*Levier	4	34087	8521,8	23,55	0,000
Retard*Champs d application*Exploration	2	17770	8884,9	24,56	0,000
Retard*Levier*Exploration	4	26790	6697,5	18,51	0,000
Champs d application*Levier*Exploration	2	731	365,4	1,01	0,367
Interactions à 4 facteur(s)	4	8564	2141,1	5,92	0,000
Retard*Champs d application*Levier*Exploration	4	8564	2141,1	5,92	0,000
Erreur	144	52099	361,8		
Total	179	339038			
Récapitulatif du modèle					
	S	R carré	R carré (ajust)	R carré (prév)	
	19,0210	84,63%	80,90%	75,99%	
R : Valeur résiduelle élevée					

Figure 40: Analyse statistique de la variance.

Le graphique (Figure 39) montre que toutes les interactions d'ordre 2 sont significatives l'analyse de la variance confirme cette observation et montre que des interactions d'ordre 3 peuvent aussi être influentes. Ainsi si la méthode d'exploration n'a pas un impact significatif sur la durée de résolution, nous remarquons que celle-ci interagit fortement avec la position du retard. Par exemple, pour la position de retard t_{64} (début de ligne), le choix de la méthode d'exploration influence le temps de résolution de manière forte : la recherche en profondeur aléatoire, « random depth », ajoute 26 s et la recherche en largeur, « breadth », diminue le temps de 15 s. D'une manière générale, l'observation de l'effet « position du retard x exploration » montre que le parcours en largeur est plus stable que le parcours en profondeur.

Nous pouvons tirer de cette analyse que que plus le nombre de possibilités est élevé, plus la résolution est longue. En effet, si le retard survient sur le premier poste, de nombreuses tâches sont disponibles pour une réallocation, alors que s'il apparaît sur un autre poste, ce nombre sera plus faible. Pour les leviers d'action, lorsque le nombre de tâches disponibles est fixé, la variation du nombre de possibilités dépend du nombre de leviers d'action activés.

Nous remarquons aussi que la méthode d'exploration n'a pas un impact significatif sur la durée de résolution lorsqu'il est considéré seul. En effet, selon les cas, la méthode de recherche en largeur est

équivalente à la méthode de recherche en profondeur relancée plusieurs fois (pour parcourir une plus grande portion de l'espace d'états).

De même, nous pouvons aussi remarquer que l'interaction entre le champ d'application et le levier d'action choisi est forte. Cette information est intéressante et mérite d'être analysée sur un plus grand nombre d'exemples afin d'en déduire des règles de résolution en fonction de la position du retard détecté.

Suite à l'analyse de la variance nous pouvons aussi conclure que la valeur résiduelle élevée est peut-être due à des facteurs de bruits (facteur cachés). Par exemple, sur le diagramme d'interaction entre la position du retard et le champ d'application, nous pouvons voir des similitudes. Leurs impacts sont similaires car ils contribuent à un facteur bruit commun : la marge globale. Lorsqu'un champ d'application restreint est sélectionné, la marge globale du problème considéré est réduite car la réallocation sur certains postes est interdite, privant alors la ligne de leur marge respective (Tableau 15). Cependant, lorsque le retard survient sur l'un des derniers postes, le choix d'un champ d'application n'impacte pas la résolution car la restriction imposée par la position du retard est au moins aussi forte.

	Cas 1	Cas 2	Cas 3
Marge totale des postes disponibles	261	192	35
Marge du poste en retard et du poste le moins chargé	-5	121	35

Tableau 15: Les marges des postes.

Une analyse plus poussée de ces facteurs bruits permettra d'améliorer l'utilisation de nos solutions de résolution.

V.3.2. Analyse des effets des facteurs sur le nombre de changements

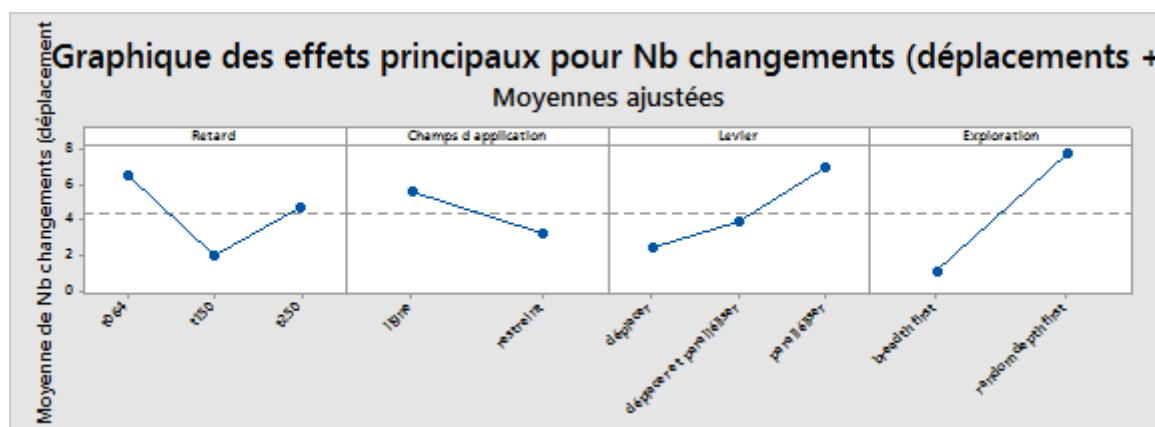


Figure 41: Graphique des effets principaux sur la durée de résolution.

Par ailleurs, nous pouvons aussi observer le diagramme des effets principaux pour le nombre de changements (Figure 41). Nous remarquons alors que le nombre de changements est plus élevé lorsque le champ d'application correspond à l'ensemble de la ligne. Cela est dû au fait que le nombre de tâches et de postes est plus élevés, donc le nombre de mouvements possibles aussi.

Nous remarquons aussi que le levier de parallélisation a tendance à toujours déplacer plus que les autres leviers. Cela est dû au fait que le déplacement utilise la marge disponible alors que la parallélisation peut être appliquée autant de fois que nécessaire, car aucun coût ne lui est associé.

Nous remarquons aussi que la recherche aléatoire génère des solutions comportant plus de déplacements que la recherche en largeur. En effet, la recherche en largeur teste d'abord les différentes possibilités à un mouvement avant de tester celles à deux mouvements. Alors que la profondeur applique plusieurs leviers.

VI. Conclusion

Ces différents cas d'étude ont permis d'illustrer et de valider la méthode. En effet, celle-ci permet de s'adapter à différentes lignes et d'adapter le modèle pour prendre en compte les différents leviers en restreignant leurs champs d'application.

L'analyse d'un cas d'étude plus complexe a permis de quantifier les valeurs de résolution. En plus de cette quantification, l'analyse a mis en valeur les interactions entre les différents éléments du problème et le besoin de prendre en compte d'autres facteurs.

Les graphiques des effets et les données de l'analyse de la variance montrent que les interactions sont très influentes, ce qui rend l'analyse par plan d'expériences pertinente. Comme perspective, il serait intéressant d'améliorer cette analyse en réalisant plus de tests pour améliorer la qualité des interprétations et en incluant d'autres facteurs, en particulier ceux liés à l'équilibrage initial (la marge des postes, le nombre de contraintes, etc.). Cependant, en l'état, nous avons pu en tirer quelques conclusions.

Conclusion générale et perspectives

Pour résoudre le problème de la modélisation et de la résolution d'un rééquilibrage dynamique des lignes d'assemblage, deux méthodes ont été étudiées : la méta-heuristique ILS et une méthode utilisant la recherche d'atteignabilité sur un réseau d'automates représentant les tâches et les postes de travail de la ligne.

Cette étude a montré que ces deux méthodes étaient difficiles à adapter à l'application de nouveaux leviers d'actions. Une approche méthodologique a donc été proposée. Celle-ci est basée sur une recherche d'atteignabilité sur un réseau d'automates communicants qui, cette fois, modélisent les différents leviers d'actions qui permettent de modifier l'équilibrage de la ligne. L'originalité de l'approche est de proposer des opérateurs élémentaires de résolution permettant une meilleure ré-utilisabilité tout le long du cycle de vie de la méthodologie.

Cette méthode de résolution a été couplée au système d'information de l'entreprise afin de pouvoir lancer la recherche de solutions lorsqu'un retard est détecté et d'appliquer le nouvel équilibrage lorsque celui-ci est trouvé. Plusieurs expériences ont été menées selon une approche structurée de type plan d'expériences, nous permettant ainsi de démontrer sa validité. Nous aurons aussi la possibilité, avec une meilleure analyse des facteurs influents et la conduite d'expériences supplémentaires, de proposer une heuristique globale de résolution. Cette heuristique aura pour objectif de fixer les paramètres de résolution selon le cas étudié afin d'optimiser le temps de résolution mais aussi de minimiser le nombre de changements.

Une autre piste pour améliorer le temps de résolution est de réduire le risque d'explosion combinatoire. Pour cela, il faut utiliser des algorithmes de parcours de l'espace d'états disposant d'une intelligence en utilisant les symétries ou les similitudes (comme par exemple la réduction d'ordre partiel de (Peled, 1993)) contenues dans le modèle pour réduire le nombre d'explorations. Les créateurs d'UPPAAL envisagent l'implémentation d'un de ces algorithmes afin de toujours améliorer leur outil de vérification formelle.

Néanmoins, nous sommes conscients que la proposition d'une méthodologie est un travail de longue haleine, c'est pourquoi, dans l'état actuel, elle est encore incomplète. En effet, elle ne tient pas compte de toutes les contraintes et de tous les leviers possibles dans notre réalité industrielle. Une des perspectives est donc d'enrichir la méthodologie par l'ajout d'autres modules élémentaires. En effet, pour cette première approche, certains éléments n'ont pas été modélisés. Cela constitue des perspectives d'amélioration de la méthode afin de s'approcher des cas industriels qui sont toujours plus complexes. Par exemple pour paralléliser des tâches, il faut amener au poste concerné un opérateur supplémentaire. Dans ce cas, il faut vérifier la disponibilité de ceux-ci avant d'allouer une tâche. Les opérateurs sont répartis sur la ligne en lien avec la répartition des tâches. Ce problème d'assignation est déjà traité dans la littérature, mais pas encore au problème de rééquilibrage dynamique. De même la parallélisation engendre un coût. Afin de le traiter de manière plus précise qu'avec l'heuristique de reconfiguration globale, un coût pourrait être assigné à chaque utilisation d'un levier afin de rechercher une réallocation dont le coût n'est pas trop élevé (la vitesse de résolution l'emporte encore sur l'optimalité) et chercher celle dont le coût est minimum.

Enfin même si l'application directe de la méthode n'a pas encore été faite dans les usines de l'entreprise, nous avons participé à l'amélioration du système d'information de celle-ci afin de pouvoir accueillir des approches de pilotage comme le rééquilibrage dynamique. Le but étant alors d'assurer la récupération d'informations terrains avant le lancement de la résolution, afin de permettre l'application de la solution trouvée. Ainsi, à ce jour, des outils de suivi du takt time ont été mis en place sur les lignes de production de l'entreprise Trane. Ceux-ci permettent aux gestionnaires de la ligne d'avoir rapidement une vision de la position des retards de leur ligne, mais aussi de conserver la trace des différentes valeurs du « takt attainment ». Cette application est utilisée sur l'ensemble des lignes des usines de Charmes et Golbey. Nous avons aussi travaillé sur l'outil d'OMS online qui permet de diffuser la liste des tâches qui doivent être réalisées à chaque poste de travail (OMS, Operational Method Sheet) de manière rapide par mise à jour informatique. Cette application est de plus en plus utilisée dans les différentes usines du groupe.

Nous avons aussi montré par la simulation, que lorsque le nombre de rééquilibrages dynamiques devenait trop grand, l'équilibrage global de la ligne devait être remis en cause. Celui-ci peut être recalculé par une méthode de Programmation Linéaire en Nombres Entiers afin de calculer une nouvelle solution qui soit optimale. Le mode prédictif et le mode réactif travaillent ainsi de manière hybride avec le même objectif : assurer au mieux la production de la ligne d'assemblage. Dans ce cas, chaque méthode est utilisée au mieux de sa capacité. Pour le prédictif, nous cherchons à atteindre l'optimalité quitte à attendre plus longtemps, alors que pour le réactif nous privilégions la rapidité de résolution. L'amélioration de ce couplage est alors une autre perspective. Cela consiste à intégrer la méthode de résolution au sein du système d'information, celui-ci détectant les retards et proposant, en plus de l'alerte, une possible solution pouvant rapidement être appliquée.

Bibliographie

- Alexander, C., Ishikawa, S., Silverstein, M., 1977. A pattern language: towns, buildings, construction. Oxford University Press.
- Altemeier, S., Helmdach, M., Koberstein, A., Dangelmaier, W., 2010. Reconfiguration of assembly lines under the influence of high product variety in the automotive industry—a decision support system. *International Journal of Production Research* 48, 6235–6256. doi:10.1080/00207540903252290
- Alur, R., Dill, D.L., 1994. A theory of timed automata. *Theoretical Computer Science* 126, 183–235. doi:10.1016/0304-3975(94)90010-8
- Amen, M., 2000. Heuristic methods for cost-oriented assembly line balancing: A survey. *International Journal of Production Economics* 68, 1–14. doi:10.1016/S0925-5273(99)00095-X
- Araoui, K., Alpan, G., Frein, Y., 2013. Minimisation des retards dans le séquençement des véhicules sur une ligne d'assemblage multimodèle. *Journal Européen des Systèmes Automatisés* 47, 635–655. doi:10.3166/jesa.47.635-655
- Arcus, A.L., 1965. A Computer Method of Sequencing Operations for Assembly Lines. *International Journal of Production Research* 4, 259–277. doi:10.1080/00207546508919982
- Bard, J.F., 1989. Assembly line balancing with parallel workstations and dead time. *International Journal of Production Research* 27, 1005–1018. doi:10.1080/00207548908942604
- Battaïa, O., Dolgui, A., 2013. A taxonomy of line balancing problems and their solution approaches. *International Journal of Production Economics* 142, 259–277. doi:10.1016/j.ijpe.2012.10.020
- Bautista, J., Pereira, J., 2007. Ant algorithms for a time and space constrained assembly line balancing problem. *European Journal of Operational Research* 177, 2016–2032. doi:10.1016/j.ejor.2005.12.017
- Baybars, İ., 1986. A Survey of Exact Algorithms for the Simple Assembly Line Balancing Problem. *Management Science* 32, 909–932. doi:10.1287/mnsc.32.8.909
- Becker, C., Scholl, A., 2006. A survey on problems and methods in generalized assembly line balancing. *European Journal of Operational Research* 168, 694–715. doi:10.1016/j.ejor.2004.07.023
- Behrmann, G., Bengtsson, J., David, A., Larsen, K.G., Pettersson, P., Yi, W., 2002. Uppaal Implementation Secrets, in: Damm, W., Olderog, E.-R. (Eds.), *Formal Techniques in Real-Time and Fault-Tolerant Systems*, Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 3–22.
- Behrmann, G., Brinksma, E., Hendriks, M., Mader, A., 2005. Production Scheduling by Reachability Analysis - A Case Study, in: *Parallel and Distributed Processing Symposium, 2005. Proceedings. 19th IEEE International. Presented at the Parallel and Distributed Processing Symposium, 2005. Proceedings. 19th IEEE International*, p. 140a–140a. doi:10.1109/IPDPS.2005.363
- Behrmann, G., David, A., Larsen, K.G., 2004. A Tutorial on Uppaal, in: Bernardo, M., Corradini, F. (Eds.), *Formal Methods for the Design of Real-Time Systems*, Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 200–236.
- Bellman, R., 1952. On the Theory of Dynamic Programming. *Proceedings of the National Academy of Sciences of the United States of America* 38, 716–719.

- Belmokhtar, S., 2006. Lignes d'usinage avec équipements standard : modélisation, configuration et optimisation. Ecole Nationale Supérieure des Mines de Saint-Etienne.
- Berard, B., Bidoit, M., Finkel, A., Laroussinie, F., Petit, A., Petrucci, L., Schnoebelen, P., 2001. Systems and Software Verification: Model-Checking Techniques and Tools, 1st ed. Springer Publishing Company, Incorporated.
- Billaut, J.-C., Moukrim, A., Sanlaville, E., 2005. Flexibilité et robustesse en ordonnancement. Hermès Science publications, Paris.
- Boysen, N., Fliedner, M., Scholl, A., 2008. Assembly line balancing: Which model to use when? International Journal of Production Economics, Special Section on Sustainable Supply Chain 111, 509–528. doi:10.1016/j.ijpe.2007.02.026
- Boysen, N., Fliedner, M., Scholl, A., 2007. A classification of assembly line balancing problems. European Journal of Operational Research 183, 674–693. doi:10.1016/j.ejor.2006.10.010
- Bukchin, Y., Rabinowitch, I., 2006. A branch-and-bound based solution approach for the mixed-model assembly line-balancing problem for minimizing stations and task duplication costs. European Journal of Operational Research 174, 492–508. doi:10.1016/j.ejor.2005.01.055
- Buxey, 1973. Production flow line system design—A review.
- Calvez, J.-P., 1990. Spécification et conception des systèmes: une méthodologie. Masson.
- Capacho, L., Pastor, R., Dolgui, A., Guschinskaya, O., 2007. An evaluation of constructive heuristic methods for solving the alternative subgraphs assembly line balancing problem. J Heuristics 15, 109–132. doi:10.1007/s10732-007-9063-x
- Castel, C., Seguin, C., 2001. Modèles formels pour l'évaluation de la sûreté de fonctionnement des architectures logicielles d'avionique modulaire intégrée. Presented at the Approches formelles dans l'assistance au développement de logiciels. Atelier, pp. 83–97.
- Chiang, W.-C., 1998. The application of a tabu search metaheuristic to the assembly line balancing problem. Annals of Operations Research 77, 209–227. doi:10.1023/A:1018925411397
- Chica, M., Cordon, Ó., Damas, S., Bautista, J., 2013. A robustness information and visualization model for time and space assembly line balancing under uncertain demand. International Journal of Production Economics 145, 761–772. doi:10.1016/j.ijpe.2013.05.030
- Chutima, P., Chimklai, P., 2012. Multi-objective two-sided mixed-model assembly line balancing using particle swarm optimisation with negative knowledge. Computers & Industrial Engineering 62, 39–55. doi:10.1016/j.cie.2011.08.015
- Clarke, E.M., Emerson, E.A., 1982. Design and synthesis of synchronization skeletons using branching time temporal logic, in: Kozen, D. (Ed.), Logics of Programs, Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 52–71.
- Corominas, A., Pastor, R., Plans, J., 2008. Balancing assembly line with skilled and unskilled workers. Omega, A Special Issue Dedicated to the 2008 Beijing Olympic Games 36, 1126–1132. doi:10.1016/j.omega.2006.03.003
- Costanza, J., 1996. Just-In-Time manufacturing excellence John Costanza Institute of Technology Inc. ISBN: B0006ENXLI.
- Denis, B., 1994. Assistance a la conception et a l'évaluation de l'architecture de conduite des systèmes de production complexes.

- Dolgui, A., Finel, B., Guschinskaya, O., Guschinsky, N., Levin, G., Vernadat, F., 2006. Balancing large-scale machining lines with multi-spindle heads using decomposition. *International Journal of Production Research* 44, 4105–4120. doi:10.1080/00207540600632232
- Dolgui, A., Finel, B., Vernadat, F., Guschinsky, N., Levin, G., 2005. A heuristic approach for transfer lines balancing. *J Intell Manuf* 16, 159–172. doi:10.1007/s10845-004-5886-6
- Dolgui, A., Guschinsky, N., Levin, G., Proth, J.-M., 2008. Optimisation of multi-position machines and transfer lines. *European Journal of Operational Research* 185, 1375–1389. doi:10.1016/j.ejor.2006.03.069
- Dolgui, A., Proth, J.-M., 2006. *Les systèmes de production modernes*. Hermès science publications : Lavoisier, Paris.
- Dreyfus, S.E., Law, A.M., 1977. *Art and Theory of Dynamic Programming*. Academic Press, Inc., Orlando, FL, USA.
- Easton, F., Faaland, B., Klastorn, T.D., Schmitt, T., 1989. Improved network based algorithms for the assembly line balancing problem. *International Journal of Production Research* 27, 1901–1915. doi:10.1080/00207548908942663
- El-Haouzi, H., 2008. *Approche méthodologique pour l'intégration des systèmes contrôlés par le produit dans un environnement de juste-à-temps: Application à l'entreprise TRANE (phdthesis)*. Université Henri Poincaré - Nancy I.
- ENV-12204, 1996. *Advanced manufacturing technology – systems architecture – constructs for entreprise modelling*, CEN TC310 WG1.
- Erel, E., Gokcen, H., 1999. Shortest-route formulation of mixed-model assembly line balancing problem. *European Journal of Operational Research* 116, 194–204. doi:10.1016/S0377-2217(98)00115-5
- Erel, E., Sarin, S.C., 1998. A survey of the assembly line balancing procedures. *Production Planning & Control* 9, 414–434. doi:10.1080/095372898233902
- Estellon, B., Gardi, F., Nouioua, K., 2008. Two local search approaches for solving real-life car sequencing problems. *European Journal of Operational Research* 191, 928–944. doi:10.1016/j.ejor.2007.04.043
- Falkenauer, E., 2005. Line balancing in the real world. Presented at the PLM'05 : international conference on product life cycle management, pp. 360–370.
- Falkenauer, E., 1992. The grouping genetic algorithms-widening the scope of the GAs. *Belgian Journal of Operations Research, Statistics and Computer Science* 33, 2.
- Frachet, J.-P., 1987. *Une introduction au génie automatique: faisabilité d'une chaîne intégrée d'outils CAO pour la conception et l'exploitation des machines automatiques industrielles*.
- Gamberini, R., Gebennini, E., Grassi, A., Regattieri, A., 2009. A multiple single-pass heuristic algorithm solving the stochastic assembly line rebalancing problem. *International Journal of Production Research* 47, 2141–2164. doi:10.1080/00207540802176046
- Gamberini, R., Grassi, A., Rimini, B., 2006. A new multi-objective heuristic algorithm for solving the stochastic assembly line re-balancing problem. *International Journal of Production Economics* 102, 226–243. doi:10.1016/j.ijpe.2005.02.013
- Ghosh, S., Gagnon, R.J., 1989. A comprehensive literature review and analysis of the design, balancing and scheduling of assembly systems. *International Journal of Production Research* 27, 637–670. doi:10.1080/00207548908942574

- Giovannini, A., 2015. A knowledge representation framework for the design and the evaluation of a product variety (phdthesis). Université de Lorraine.
- Glover, F., 1990. Tabu Search—Part II. *ORSA Journal on Computing* 2.
- Glover, F., 1989. Tabu Search—Part I. *ORSA Journal on Computing* 1.
- Graham, R.L., Lawler, E.L., Lenstra, J.K., Kan, A.H.G.R., 1979. Optimization and Approximation in Deterministic Sequencing and Scheduling: a Survey, in: P.L. Hammer, E.L.J. and B.H.K. (Ed.), *Annals of Discrete Mathematics, Discrete Optimization II Proceedings of the Advanced Research Institute on Discrete Optimization and Systems Applications of the Systems Science Panel of NATO and of the Discrete Optimization Symposium Co-Sponsored by IBM Canada and SIAM Banff, Aha. and Vancouver*. Elsevier, pp. 287–326.
- Grangeon, N., Leclaire, P., Norre, S., 2011. Heuristics for the re-balancing of a vehicle assembly line. *International Journal of Production Research* 49, 6609–6628. doi:10.1080/00207543.2010.539025
- Gutjahr, A.L., Nemhauser, G.L., 1964. An Algorithm for the Line Balancing Problem. *Management Science* 11, 308–315. doi:10.1287/mnsc.11.2.308
- Gzara, L., 2000. Les patterns pour l'ingenierie des systemes d'information produit. INP Grenoble.
- Hackman, S.T., Magazine, M.J., Wee, T.S., 1989. Fast, effective algorithms for simple assembly line balancing problems. *Operations Research* 37, 916–924.
- Hall, R.W., 1987. Attaining manufacturing excellence: just-in-time, total quality, total people involvement. Irwin Professional Pub.
- Hammer, M., Knapp, A., Merz, S., 2005. Truly On-the-Fly LTL Model Checking, in: Halbwachs, N., Zuck, L.D. (Eds.), *Tools and Algorithms for the Construction and Analysis of Systems, Lecture Notes in Computer Science*. Springer Berlin Heidelberg, pp. 191–205.
- Heinrici, A., 1994. A Comparison between Simulated Annealing and Tabu Search with an Example from the Production Planning, in: *Operations Research Proceedings 1993, Operations Research Proceedings*. Springer Berlin Heidelberg, pp. 498–503.
- Held, M., Karp, R.M., 1962. A Dynamic Programming Approach to Sequencing Problems. *Journal of the Society for Industrial and Applied Mathematics* 10, 196–210.
- Held, M., Karp, R.M., Shareshian, R., 1963. Assembly-Line Balancing-Dynamic Programming with Precedence Constraints. *Operations Research* 11, 442–459.
- Helgeson, W.B., Birnie, D.P., 1961a. Assembly line balancing using the ranked positional weight technique. *Journal of Industrial Engineering* 12, 394–398.
- Helgeson, W.B., Birnie, D.P., 1961b. Assembly line balancing using the ranked positional weight technique. *Journal of Industrial Engineering* 12, 394–398.
- Hoffmann, T.R., 1992. Eureka: A Hybrid System for Assembly Line Balancing. *Management Science* 38, 39–47.
- Jackson, J.R., 1956. A Computing Procedure for a Line Balancing Problem. *Management Science* 2, 261–271.
- Johnson, R.V., 1988. Optimally Balancing Large Assembly Lines with “Fable.” *Management Science* 34, 240–253.
- Kalayci, C.B., Gupta, S.M., 2013. A particle swarm optimization algorithm with neighborhood-based mutation for sequence-dependent disassembly line balancing problem. *Int J Adv Manuf Technol* 69,

197–209. doi:10.1007/s00170-013-4990-1

- Kim, D., Park, J., Moon, I., 2015. Integrated Mixed-Model Assembly Line Balancing with Unskilled Temporary Workers, in: Umeda, S., Nakano, M., Mizuyama, H., Hibino, H., Kiritsis, D., Cieminski, G. von (Eds.), *Advances in Production Management Systems: Innovative Production Management Towards Sustainable Growth*, IFIP Advances in Information and Communication Technology. Springer International Publishing, pp. 324–331.
- Kim, Y.K., Kim, Y.J., Kim, Y., 1996. Genetic algorithms for assembly line balancing with various objectives. *Computers & Industrial Engineering*, IE in Korea 30, 397–409. doi:10.1016/0360-8352(96)00009-5
- Klein, R., Scholl, A., 1996. Maximizing the production rate in simple assembly line balancing — A branch and bound procedure. *European Journal of Operational Research* 91, 367–385. doi:10.1016/0377-2217(95)00047-X
- Ko, J., Hu, S.J., Huang, T., 2005. Reusability Assessment for Manufacturing Systems. *CIRP Annals - Manufacturing Technology* 54, 113–116. doi:10.1016/S0007-8506(07)60062-6
- Koren, Y., Heisel, U., Jovane, F., Moriwaki, T., Pritschow, G., Ulsoy, G., Van Brussel, H., 1999. Reconfigurable Manufacturing Systems. *CIRP Annals - Manufacturing Technology* 48, 527–540. doi:10.1016/S0007-8506(07)63232-6
- Kottas, J.F., Lau, H.-S., 1973. A Cost-Oriented Approach to Stochastic Line Balancing1. *A I I E Transactions* 5, 164–171. doi:10.1080/05695557308974897
- Land, A.H., Doig, A.G., 1960. An Automatic Method of Solving Discrete Programming Problems. *Econometrica* 28, 497–520. doi:10.2307/1910129
- Lapierre, S.D., Ruiz, A.B., 2004. Balancing assembly lines: an industrial case study. *Journal of the Operational Research Society* 55, 589–597. doi:10.1057/palgrave.jors.2601708
- Lapierre, S.D., Ruiz, A., Soriano, P., 2006. Balancing assembly lines with tabu search. *European Journal of Operational Research*, Balancing Assembly and Transfer lines 168, 826–837. doi:10.1016/j.ejor.2004.07.031
- Lawler, E.L., Wood, D.E., 1966. Branch-And-Bound Methods: A Survey. *Operations Research* 14, 699–719.
- Lematre, T., 2013. Allocation de fonctions de commande de systèmes critiques par recherche d'atteignabilité dans un réseau d'automates communicants. École normale supérieure de Cachan - ENS Cachan.
- Lesert, A., Alpan, G., Frein, Y., Noiré, S., 2011. Definition of spacing constraints for the car sequencing problem. *International Journal of Production Research* 49, 963–994. doi:10.1080/00207540903469043
- Lesert, A., Alpan, G., Frein, Y., Noiré, S., 2010. Evaluation de la flexibilité des postes de travail dans une usine terminale automobile. *Logistique & Management* 18, 57–68.
- Makssoud, F., 2014. Apport de l'optimisation combinatoire pour la reconfiguration des lignes de production (phdthesis). Ecole Nationale Supérieure des Mines de Saint-Etienne.
- Makssoud, F., Battaïa, O., Dolgui, A., 2014. An exact optimization approach for a transfer line reconfiguration problem. *Int J Adv Manuf Technol* 72, 717–727. doi:10.1007/s00170-014-5694-x
- Makssoud, F., Battaïa, O., Dolgui, A., 2013. Reconfiguration of Machining Transfer Lines, in: Borangiu, T., Thomas, A., Trentesaux, D. (Eds.), *Service Orientation in Holonic and Multi Agent Manufacturing and Robotics*, Studies in Computational Intelligence. Springer Berlin Heidelberg, pp. 339–353.

- Manceaux, A., Bril El-Haouzi, H., Thomas, A., Pétin, J.-F., 2014a. Dynamic Rebalancing of an Assembly Line with a Reachability Analysis of Communicating Automata, in: Grabot, B., Vallespir, B., Gomes, S., Bouras, A., Kiritsis, D. (Eds.), *Advances in Production Management Systems. Innovative and Knowledge-Based Production Management in a Global-Local World*, IFIP Advances in Information and Communication Technology. Springer Berlin Heidelberg, pp. 597–604.
- Manceaux, A., Bril El-Haouzi, H., Thomas, A., Pétin, J.-F., Munier, T., 2014b. Reconfiguration dynamique d'une ligne d'assemblage par recherche d'atteignabilité sur un réseau d'automates communicants, in: 10ème Conférence Francophone de Modélisation, Optimisation et Simulation, MOSIM'14. Nancy, France.
- Marangé, P., Pétin, J.-F., Manceaux, A., Gouyon, D., 2011. Contribution à la reconfiguration des systèmes de production. *Journal Européen des Systèmes Automatisés* 45, 45–60. doi:10.3166/jesa.45.45-60
- McMullen, P.R., Frazier, G.V., 1997. A heuristic for solving mixed-model line balancing problems with stochastic task durations and parallel stations. *International Journal of Production Economics* 51, 177–190. doi:10.1016/S0925-5273(97)00048-0
- Mehrabi, M.G., Ulsoy, A.G., Koren, Y., 2000. Reconfigurable manufacturing systems: Key to future manufacturing. *Journal of Intelligent Manufacturing* 11, 403–419. doi:10.1023/A:1008930403506
- Meinadier, J.-P., 2002. *Le métier d'intégration de système*. Hermes.
- Miralles, C., García-Sabater, J.P., Andrés, C., Cardós, M., 2008. Branch and bound procedures for solving the Assembly Line Worker Assignment and Balancing Problem: Application to Sheltered Work centres for Disabled. *Discrete Applied Mathematics* 156, 352–367. doi:10.1016/j.dam.2005.12.012
- Moodie, C.L., 1964. *A Heuristic Method of Assembly Line Balancing for Assumptions of Constant or Variable Work Element Times* (phdthesis). Purdue University.
- Morand, B., 1995. Statut épistémologique des modèles dans la conception des systèmes d'information. *Ingénierie des systèmes d'information* 3, 665–700.
- Morel, G., 1992. *Contribution à l'automatisation et à l'ingénierie des systèmes intégrés de production*. Habilitation à Diriger des Recherches de l'Université Henri Poincaré 28.
- Nourie, F.J., Venta, E.R., 1991. Finding optimal line balances with OptPack. *Operations Research Letters* 10, 165–171. doi:10.1016/0167-6377(91)90034-M
- Özcan, U., Toklu, B., 2008. A tabu search algorithm for two-sided assembly line balancing. *Int J Adv Manuf Technol* 43, 822–829. doi:10.1007/s00170-008-1753-5
- Peled, D., 1993. All from one, one for all: on model checking using representatives, in: Courcoubetis, C. (Ed.), *Computer Aided Verification, Lecture Notes in Computer Science*. Springer Berlin Heidelberg, pp. 409–423.
- Penna, P.H.V., Subramanian, A., Ochi, L.S., 2011. An Iterated Local Search heuristic for the Heterogeneous Fleet Vehicle Routing Problem. *J Heuristics* 19, 201–232. doi:10.1007/s10732-011-9186-y
- Pierreval, H., Caux, C., Paris, J.L., Viguier, F., 2003. Evolutionary approaches to the design and organization of manufacturing systems. *Computers & Industrial Engineering* 44, 339–364. doi:10.1016/S0360-8352(02)00195-X
- Rachamadugu, R., Talbot, B., 1991. Improving the equality of workload assignments in assembly lines. *International Journal of Production Research* 29, 619–633. doi:10.1080/00207549108930092
- Rashboogari, N.S., Kermanshah, I., 2012. Solving Assembly Line Balancing Problem in the State of Multiple-

- Alternative Operations Sequence using GRASP Meta-Heuristic Method and Comparing with Genetic Algorithm. Proceedings of the 2012 International Conference on Industrial Engineering and Operations Management.
- Rekiek, B., Dolgui, A., Delchambre, A., Bratcu, A., 2002. State of art of optimization methods for assembly line design. *Annual Reviews in Control* 26, 163–174. doi:10.1016/S1367-5788(02)00027-5
- Riera, B., Annebicque, D., Vigario, B., Gellot, F., Philippot, A., others, 2012. Synthèse de commande sûre de fonctionnement à base de contraintes logiques pour les systèmes manufacturiers. Proceedings of MOSIM'12.
- Rieu, D., Giraudin, J.P., Saint-Marcel, C., 1999. Réutilisation et patrons d'ingénierie, in: *Génie Objet: Analyse et Conception de L'évolution*. Hermès Science publications.
- Sakoe, H., Chiba, S., 1978. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing* 26, 43–49. doi:10.1109/TASSP.1978.1163055
- Salveson, M.E., 1955. The assembly line balancing problem. *Journal of Industrial Engineering* 6, 18–25.
- Schnoebelen, P., Bérard, B., Bidoit, M., Laroussinie, F., 1999. Vérification de logiciels: techniques et outils du model-checking. Vuibert.
- Scholl, A., 1999. Balancing and sequencing of assembly lines. Physica-Verlag.
- Scholl, A., Becker, C., 2006. State-of-the-art exact and heuristic solution procedures for simple assembly line balancing. *European Journal of Operational Research* 168, 666–693. doi:10.1016/j.ejor.2004.07.022
- Scholl, A., Flidner, M., Boysen, N., 2010. Absalom: Balancing assembly lines with assignment restrictions. *European Journal of Operational Research* 200, 688–701. doi:10.1016/j.ejor.2009.01.049
- Scholl, A., Klein, R., 1999. Balancing assembly lines effectively – A computational comparison. *European Journal of Operational Research* 114, 50–58. doi:10.1016/S0377-2217(98)00173-8
- Scholl, A., Klein, R., 1997. SALOME: A Bidirectional Branch-and-Bound Procedure for Assembly Line Balancing. *INFORMS Journal on Computing* 9, 319.
- Scholl, A., Voß, S., 1997. Simple assembly line balancing—Heuristic approaches. *J Heuristics* 2, 217–244. doi:10.1007/BF00127358
- Schrage, L., Baker, K.R., 1978. Dynamic Programming Solution of Sequencing Problems with Precedence Constraints. *Operations Research* 26, 444–449.
- Stützle, T., 1998. Applying iterated local search to the permutation flow shop problem.
- Stützle, T., Hoos, H.H., 2002. Analysing the Run-Time Behaviour of Iterated Local Search for the Travelling Salesman Problem, in: *Essays and Surveys in Metaheuristics, Operations Research/Computer Science Interfaces Series*. Springer US, pp. 589–611.
- Subbiah, S., Engell, S., 2010. Short-Term Scheduling of Multi-Product Batch Plants with Sequence-Dependent Changeovers Using Timed Automata Models, in: S. Pierucci and G. Buzzi Ferraris (Ed.), *Computer Aided Chemical Engineering*. Elsevier, pp. 1201–1206.
- Talbot, F.B., Patterson, J.H., Gehrlein, W.V., 1986. A Comparative Evaluation of Heuristic Line Balancing Techniques. *Management Science* 32, 430–454. doi:10.1287/mnsc.32.4.430
- Thomas, A., Haouzi, H. El, Klein, T., Belmokhtar, S., Herrera, C., 2009. Architecture de systèmes contrôlés par le produit pour un environnement de juste à temps. *Journal Européen des Systèmes Automatisés*

43, 513–535. doi:10.3166/jesa.43.513-535

Vain, J., Randvee, I., Riismaa, T., Ernits, J., 2002. Solving line balancing problems with model checking. Proceedings of the Estonian Academy of Sciences, Engineering 8, 211.

Valmari, A., 1992. A stubborn attack on state explosion. Form Method Syst Des 1, 297–322. doi:10.1007/BF00709154

Valmari, A., 1989. Stubborn sets for reduced state space generation, in: Rozenberg, G. (Ed.), Advances in Petri Nets 1990, Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 491–515.

Verdanat, F., 1996. Enterprise modeling and integration: principles and application. Chapman & Hall, London.

Zavadlav, E., McClain, J.O., Thomas, L.J., 1996. Self-Buffering, Self-Balancing, Self-Flushing Production Lines. Management Science 42, 1151–1164.

Annexe 1 : Implémentation de la méthode ILS

'Enregistrement de la matrice solution initiale pour accélérer la simulation

```
Public Function Matrice_Init() As Integer()
Dim i As Integer, j As Integer
Dim Matrice() As Integer

P = Sheets("Donnees generales").Range("B4").Value
T = Sheets("Donnees generales").Range("B5").Value

ReDim Matrice(1 To P, 1 To T)

For i = 1 To T
    For j = 1 To P
        Matrice(j, i) = Sheets("Matrice initiale").Range("B2").Offset(j, i).Value
    Next
Next

Matrice_Init = Matrice

End Function
```

'Enregistrement de la matrice de précédence pour accélérer la simulation

```
Public Function Matrice_Prec() As Integer()
Dim i1 As Integer, i2 As Integer
Dim Matrice() As Integer

T = Sheets("Donnees generales").Range("B5").Value

ReDim Matrice(1 To T, 1 To T)

For i1 = 1 To T
    For i2 = 1 To T
        Matrice(i1, i2) = _
            Sheets("Matrice precedence").Range("B2").Offset(i1, i2).Value
    Next
Next

Matrice_Prec = Matrice

End Function
```

'Détermination du poste perturbé

```
Public Function Poste_Perturbe() As Integer
Dim T_Perturbee As Integer

P = Sheets("Donnees generales").Range("B4").Value
T = Sheets("Donnees generales").Range("B5").Value

T_Perturbee = Range("B11").Value

Poste_Perturbe = 1

While Sheets("Matrice initiale").Range("B2").Offset(Poste_Perturbe, T_Perturbee) = 0
    Poste_Perturbe = Poste_Perturbe + 1
Wend

End Function
```

```
'Vérification de l'influence de la perturbation
Public Function Verification_takt_time(Poste_Perturbe As Integer) As Boolean
Dim i As Integer
Dim T As Integer
Dim Duree_Poste As Integer
Dim TT As Integer

Verification_takt_time = True

TT = Range("B2").Value
T = Sheets("Donnees generales").Range("B5").Value

For i = 1 To T
    Duree_Poste = _
    Duree_Poste + _
    Sheets("Taches").Range("B1").Offset(i, 0).Value * _
    Sheets("Matrice initiale").Range("B2").Offset(Poste_Perturbe, i).Value
Next
Duree_Poste = Duree_Poste + Sheets("Donnees generales").Range("C11").Value

If Duree_Poste <= TT Then
    Verification_takt_time = False
End If

End Function
```

' Fonction vérifiant si la tâche perturbée est le dernier du poste ou non.

```
Public Function Derniere_Tache() As Boolean
Dim T_Perturbee As Integer
Dim MC1 As Range, MC2 As Range, MC3 As Range
Dim i As Integer, j As Integer

P = Sheets("Donnees generales").Range("B4").Value
T = Sheets("Donnees generales").Range("B5").Value

T_Perturbee = Range("B11").Value

Set MC1 = Sheets("Assignation initiale").Range("B1")
Set MC2 = Sheets("Assignation initiale").Cells(P + 1, T + 2)

If Sheets("Assignation initiale").Range(MC1, MC2).Find(T_Perturbee).Offset(1, 0) _
= 0 Then
    Derniere_Tache = True
Else
    Derniere_Tache = False
End If

End Function
```

```

'Détermine si le nouveau temps de cycle optimal est inférieur au takt time
Public Function Temps_cycle_optimal(Poste_Perturbe) As Double
Dim TT As Integer
Dim i As Integer, j As Integer
Dim Temps_Total As Integer

TT = Range("B2").Value
P = Sheets("Donnees generales").Range("B4").Value
T = Sheets("Donnees generales").Range("B5").Value

Temps_Total = 0

For j = Poste_Perturbe To P
    For i = 1 To T
        Temps_Total = Temps_Total + _
            Sheets("Taches").Range("B1").Offset(i, 0).Value * _
            Sheets("Matrice initiale").Range("B2").Offset(j, i).Value
    Next
Next

'Ajout de la perturbation
Temps_Total = Temps_Total + Sheets("Donnees generales").Range("C11").Value

Temps_cycle_optimal = Temps_Total / (P - Poste_Perturbe + 1)

End Function

```

```

'Tableau des tâches du poste perturbé
Public Function Taches_Poste_Pert(T_Perturbee, P_Perturbe) As Integer()
Dim i As Integer
Dim MC1 As Range
Dim T As Integer
Dim Tab1() As Integer

T = Sheets("Donnees generales").Range("B5").Value

Set MC1 = Sheets("Assignment initiale").Range("C1").Offset(P_Perturbe, 0)

Set MC1 = Sheets("Assignment initiale").Range(MC1, _
MC1.Offset(0, T)).Find(T_Perturbee)

ReDim Tab1(1 To Sheets("Assignment initiale").Range("A1"). _
Offset(P_Perturbe, 1).Value + 2 - MC1.Column, 1 To 4)

i = 1
While MC1.Offset(0, i) <> 0
    Tab1(i, 1) = i
    Tab1(i, 2) = MC1.Offset(0, i).Value
    Tab1(i, 3) = Sheets("Taches").Range("B1").Offset(MC1.Offset(0, i).Value, 0).Value
    Tab1(i, 4) = P_Perturbe
    i = i + 1
Wend

Taches_Poste_Pert = Tab1

End Function

```

'Tableau des tâches en aval du poste perturbé

```
Public Function Taches_Postes_Aval(P_Perturbe As Integer) As Integer()
Dim i As Integer, i_precedent As Integer, j As Integer
Dim MC1 As Range, MC2 As Range, MC3 As Range
Dim Tabl() As Integer

T = Sheets("Donnees generales").Range("B5").Value
P = Sheets("Donnees generales").Range("B4").Value

Set MC1 = Sheets("Assignment initiale").Range("B1")
Set MC2 = MC1.Offset(P_Perturbe + 1, 0)
Set MC3 = MC1.End(xlDown)
ReDim Preserve _
Tabl(1 To Application.WorksheetFunction.Sum _
(Sheets("Assignment initiale ").Range(MC2, MC3)) + P - P_Perturbe, 1 To 4)

j = P_Perturbe + 1
i = 1
i_precedent = 1
While MC1.Offset(j, i_precedent) <> 0
    While MC1.Offset(j, i_precedent) <> 0
        Tabl(i, 1) = i
        Tabl(i, 2) = MC1.Offset(j, i_precedent).Value
        Tabl(i, 3) = _
Sheets("Taches").Range("B1").Offset(MC1.Offset(j, i_precedent).Value, 0).Value
        Tabl(i, 4) = j
        i = i + 1
        i_precedent = i_precedent + 1
    Wend
    Tabl(i, 1) = i
    Tabl(i, 2) = 0
    Tabl(i, 3) = 0
    Tabl(i, 4) = j
    i_precedent = 1
    i = i + 1
    j = j + 1
Wend

Taches_Postes_Aval = Tabl

End Function
```

```
'Creation de la matrice taboue/ matrice mémoire
Public Function Matrice_Taboue(Taches_Poste_Bloquant, _
Autres_Taches_Reassignables) As Boolean()
Dim T1 As Integer, T2 As Integer
Dim P As Integer
Dim Matrice() As Boolean

ReDim Matrice(UBound(Taches_Poste_Bloquant, 1), _
UBound(Autres_Taches_Reassignables, 1))

For T1 = 1 To UBound(Taches_Poste_Bloquant, 1)
    For T2 = 1 To UBound(Autres_Taches_Reassignables, 1)
        'Rien n'a été testé pour l'instant
        Matrice(T1, T2) = False
    Next
Next

Matrice_Taboue = Matrice

End Function
```

```
'Optimisation : utilisation de table mémoire plutôt que matrice
Public Function Table_Taboue(Taches_Poste_Bloquant) As Integer()
Dim T1 As Integer
Dim Table() As Integer

ReDim Table(1, UBound(Taches_Poste_Bloquant, 1))

For T1 = 1 To UBound(Taches_Poste_Bloquant, 1)
    Table(1, T1) = 0 'rien n'a été testé au départ
Next

Table_Taboue = Table

End Function
```

```

'Creation de la matrice mémoire 2
Public Function Matrice_Taboue2(Taches_Poste_Bloquant, Autres_Taches_Reassignables)
As Boolean()
Dim T1 As Integer, T2 As Integer, T3 As Integer
Dim P As Integer
Dim Matrice() As Boolean

ReDim Matrice(UBound(Taches_Poste_Bloquant, 1), UBound(Taches_Poste_Bloquant, 1), _
UBound(Autres_Taches_Reassignables, 1))

For T1 = 1 To UBound(Taches_Poste_Bloquant, 1)
For T2 = 1 To UBound(Taches_Poste_Bloquant, 1)
'Rien n'a été testé pour l'instant
For T3 = 1 To UBound(Autres_Taches_Reassignables, 1)
Matrice(T1, T2, T3) = False
Next
Next
Next

Matrice_Taboue2 = Matrice

End Function

```

```

'tache aléatoire
Public Function Tache_Aleatoire(Tableau) As Integer
Randomize
Tache_Aleatoire = Int(UBound(Tableau, 1) * Rnd) + 1
End Function

```

```

'Calcul du temps de cycle pour le poste perturbé
Public Function Tps_Cycle(Poste_Perturbe As Integer, Matrice) As Integer
Dim i As Integer, j As Integer
Dim TC As Integer

T = Sheets("Donnees generales").Range("B5").Value
P = Sheets("Donnees generales").Range("B4").Value

For j = Poste_Perturbe To P
TC = 0
For i = 1 To T
TC = TC + Matrice(j, i) * Sheets("Taches").Range("B1").Offset(i, 0).Value
Next
If j = Poste_Perturbe Then
TC = TC + Sheets("Donnees generales").Range("C11").Value
End If
If TC > Tps_Cycle Then
Tps_Cycle = TC
End If
Next
End Function

```

```

Public Function Test_Precedence(Matrice_Affectation, Matrice_Precedence, _
Tache As Integer) As Boolean
Dim i As Integer, j As Integer
Dim P1 As Integer, Pj As Integer

T = Sheets("Donnees generales").Range("B5").Value
P = Sheets("Donnees generales").Range("B4").Value

If Tache <> 0 Then
    'Determination du poste de la tache
    For j = 1 To P
        If Matrice_Affectation(j, Tache) = 1 Then
            P1 = j
        End If
    Next

    Test_Precedence = True

    'Test des successeurs
    For i = 1 To T
        If Matrice_Precedence(Tache, i) = 1 Then
            For j = 1 To P
                If Matrice_Affectation(j, i) = 1 Then
                    Pj = j
                End If
            Next
            If Pj < P1 Then
                Test_Precedence = False
                Exit Function
            End If
        End If
    Next
    'Test des predecesseurs
    For i = 1 To T
        If Matrice_Precedence(i, Tache) = 1 Then
            For j = 1 To P
                If Matrice_Affectation(j, i) = 1 Then
                    Pj = j
                End If
            Next
            If Pj > P1 Then
                Test_Precedence = False
                Exit Function
            End If
        End If
    Next
Else
    Test_Precedence = True
End If

End Function

```

```

'nouveau post bloquant et ses taches
Public Function Tableau_Poste_Bloquant(Matrice, P_Perturbe As Integer, _
Tableau_Poste_Bloq, Tableau_Autres_Post) As Integer()
Dim i As Integer, i_precedent As Integer, j As Integer, k As Integer
Dim MC1 As Range, MC2 As Range, MC3 As Range
Dim Tab1() As Integer
Dim Poste_Bloquant As Integer
Dim Tps_Cycle As Integer, TC As Integer
Dim Nb_taches As Integer

T = Sheets("Donnees generales").Range("B5").Value
P = Sheets("Donnees generales").Range("B4").Value
Tps_Cycle = 0

For j = P_Perturbe To P
    TC = 0
    For i = 1 To T
        TC = TC + Matrice(j, i) * Sheets("Taches").Range("B1").Offset(i, 0).Value
    Next
    If j = P_Perturbe Then
        TC = TC + Sheets("Donnees generales").Range("C11").Value
    End If
    If TC > Tps_Cycle Then
        Tps_Cycle = TC
        Poste_Bloquant = j
    End If
Next

i_precedent = 1
If Tableau_Poste_Bloq(1, 4) = Poste_Bloquant Then
    'Tableau_Poste_Bloq est tableau des taches du postes perturbé
    Tab1 = Tableau_Poste_Bloq
Else
    Nb_taches = 0
    For k = 1 To UBound(Tableau_Autres_Post, 1)
        'Tableau_Autre_Post est tableau des taches du postes en aval
        If Tableau_Autres_Post(k, 4) = Poste_Bloquant _
        And Tableau_Autres_Post(k, 3) <> 0 Then
            Nb_taches = Nb_taches + 1
            ' déterminer le nombre des taches du nouveau post bloquant
        End If
    Next
    ReDim Tab1(Nb_taches, 4)
    ' Créer le nouveau tableau des taches du post bloquant
    For k = 1 To UBound(Tableau_Autres_Post, 1)
        If Tableau_Autres_Post(k, 4) = Poste_Bloquant And _
        Tableau_Autres_Post(k, 3) <> 0 Then
            Tab1(i_precedent, 1) = i_precedent
            Tab1(i_precedent, 2) = Tableau_Autres_Post(k, 2)
            Tab1(i_precedent, 3) = Tableau_Autres_Post(k, 3)
            Tab1(i_precedent, 4) = Tableau_Autres_Post(k, 4)
            i_precedent = i_precedent + 1
        End If
    Next
End If

Tableau_Poste_Bloquant = Tab1

End Function

```

```

'Nouveau tableau des postes en aval du post bloquant
Public Function Tableau_Autres_Postes(Matrice, _
P_Perturbe As Integer, Tableau_Poste_Bloq, Tableau_Autres_Post) As Integer()
Dim i As Integer, i_precedent As Integer, j As Integer, k As Integer
Dim MC1 As Range, MC2 As Range, MC3 As Range
Dim Tab1() As Integer
Dim Poste_Bloquant As Integer
Dim Tps_Cycle As Integer, TC As Integer
Dim Nb_taches As Integer

T = Sheets("Donnees generales").Range("B5").Value
P = Sheets("Donnees generales").Range("B4").Value
Tps_Cycle = 0

For j = P_Perturbe To P
    TC = 0
    For i = 1 To T
        TC = TC + Matrice(j, i) * Sheets("Taches").Range("B1").Offset(i, 0).Value
    Next
    If j = P_Perturbe Then
        TC = TC + Sheets("Donnees generales").Range("C11").Value
    End If
    If TC > Tps_Cycle Then
        Tps_Cycle = TC
        Poste_Bloquant = j
    End If
Next

```

```

i_precedent = 1
If Tableau_Poste_Bloq(1, 4) = Poste_Bloquant Then
    Tab1 = Tableau_Autres_Post
Else
    Nb_taches = 0
    For k = 1 To UBound(Tableau_Autres_Post, 1)
        If Tableau_Autres_Post(k, 4) = Poste_Bloquant Then
            Nb_taches = Nb_taches + 1
        End If
    Next
    ReDim Tab1(UBound(Tableau_Poste_Bloq, 1) + 1 + UBound(Tableau_Autres_Post, 1) _
    - Nb_taches, 4)
    For k = 1 To UBound(Tableau_Poste_Bloq, 1)
        Tab1(i_precedent, 1) = i_precedent
        Tab1(i_precedent, 2) = Tableau_Poste_Bloq(k, 2)
        Tab1(i_precedent, 3) = Tableau_Poste_Bloq(k, 3)
        Tab1(i_precedent, 4) = Tableau_Poste_Bloq(k, 4)
        i_precedent = i_precedent + 1
    Next
    Tab1(i_precedent, 1) = i_precedent
    Tab1(i_precedent, 2) = 0
    Tab1(i_precedent, 3) = 0
    Tab1(i_precedent, 4) = Tableau_Poste_Bloq(k - 1, 4)
    i_precedent = i_precedent + 1

```

```

    For k = 1 To UBound(Tableau_Autres_Post, 1)
        If Tableau_Autres_Post(k, 4) <> Poste_Bloquant Then
            Tabl(i_precedent, 1) = i_precedent
            Tabl(i_precedent, 2) = Tableau_Autres_Post(k, 2)
            Tabl(i_precedent, 3) = Tableau_Autres_Post(k, 3)
            Tabl(i_precedent, 4) = Tableau_Autres_Post(k, 4)
            i_precedent = i_precedent + 1
        End If
    Next
End If

Tableau_Autres_Postes = Tabl

End Function

```

'Création matrice des précédences

```

Private Sub Precedence_Click()
    Dim i As Integer
    Dim P As Integer, T As Integer
    Dim j As Integer

    Sheets("Matrice precedence").Cells.Clear
    Sheets("Matrice precedence").Cells.Borders.LineStyle = xlNone

    T = Sheets("Donnees generales").Range("B5").Value

    Sheets("Matrice precedence").Range("A1") = "Tâches"
    Sheets("Matrice precedence").Range("A1").Borders.Weight = xlThin
    Sheets("Matrice precedence").Range("A1").Offset(1, 0) = "Prédécesseurs"
    Sheets("Matrice precedence").Range("A1").Offset(1, 0).Borders.Weight = xlThin
    Sheets("Matrice precedence").Range("A1").Offset(0, 1) = "Successeurs"
    Sheets("Matrice precedence").Range("A1").Offset(0, 1).Borders.Weight = xlThin

    For j = 1 To T
        Sheets("Matrice precedence").Range("A2").Offset(j, 0).Value = j
        Sheets("Matrice precedence").Range("A2").Offset(j, 0).Borders.Weight = xlThin
        Sheets("Matrice precedence").Range("B1").Offset(0, j).Value = j
        Sheets("Matrice precedence").Range("B1").Offset(0, j).Borders.Weight = xlThin
        For k = 1 To T
            Sheets("Matrice precedence").Range("B2").Offset(j, k).Value = 0
            Sheets("Matrice precedence").Range("B2").Offset(j, k).Borders.Weight = _
                xlThin
        Next
    Next

    i = 1

```

```

While IsEmpty(Sheets("Precedence").Range("A1").Offset(i, 0)) = False
    Sheets("Matrice precedence").Range("B2").Offset(Sheets("Precedence" _
        ).Range("A1").Offset(i, 0).Value, Sheets("Precedence" _
        ).Range("A1").Offset(i, 1).Value) = 1
    i = i + 1
Wend

End Sub

```

```

Private Sub CommandButton1_Click()

Dim i As Integer, j As Integer, k As Integer
Dim P As Integer, T As Integer
Dim S As Integer

P = Sheets("Donnees generales").Range("B4").Value
T = Sheets("Donnees generales").Range("B5").Value

Sheets("Matrice initiale").Cells.Clear
Sheets("Matrice initiale").Range("B1") = "Atelier"
Sheets("Matrice initiale").Range("A2") = "Poste"

' Création de la matrice initiale
For j = 1 To P
    Sheets("Matrice initiale").Range("A1").Offset(j + 1, 0).Value = j
Next

For i = 1 To T
    Sheets("Matrice initiale").Range("A1").Offset(0, i + 1).Value = i
Next

For j = 1 To P
    For i = 1 To T
        Sheets("Matrice initiale").Range("A1").Offset(j + 1, i + 1).Value = 0
    Next
Next

For j = 1 To P
    k = 1
    While IsEmpty(Sheets("Assignations initiales").Range("A1").Offset(j, _
k + 1)) = False
        Sheets("Matrice initiale").Range("B2").Offset(j, Sheets( _
"Assignations initiales").Range("A1").Offset(j, k + 1).Value).Value = 1
        k = k + 1
    Wend
    Sheets("Assignations initiales").Range("A1").Offset(j, 1).Value = k - 1
Next

' Vérifie que toutes les tâches ont bien été assignées
For i = 1 To T
    S = 0

    For j = 1 To P
        S = S + Sheets("Matrice initiale").Range("B2").Offset(j, i).Value
    Next

    If S <> 1 Then
        MsgBox ("La tâche " & i & " n'a pas été assignée")
        Exit Sub
    End If

Next

MsgBox ("Toutes les tâches ont été assignées")

End Sub

```

Annexe 2 : Implémentation de la méthode proposée

I. Script de lancement des résolutions

I.1. Initialisation : définition de la ligne

Il s'agit ici de modifier les valeurs définissant la ligne d'assemblage. Pour cela, le modèle UPPAAL au format XML est ouvert, et les valeurs sont modifiées dans la déclaration du système comme par exemple :

```
# Ouverture du fichier source
with open(template, 'r') as f:
    sourceLines = f.readlines()

#for ligne in sourceLines: #NON car on veut savoir ou on en est
i = 0
NbMaxLines = len(sourceLines)

with open(model_file_BIS, "w") as destination:

    while i < NbMaxLines:
        if "// Place global declarations here." in sourceLines[i]:
            destination.write("// Place global declarations here.\n" )
            destination.write("const int takt = %s;\n" %(takt))
            destination.write("const int No = %s;\n" %(No))
            destination.write("const int Nt = %s;\n" %(Nt))
            destination.write("int listeDesDurees[Nt] = %s\n" %(listeDesDureesTXT))
            ...
        i += 1
```

I.2. Activation des leviers

Il s'agit ici de modifier la déclaration globale du système afin d'activer les leviers désirés

```
# Modifier modele Choix des modules
if levier == "deplacer": # "deplacer" OU "paralleliser" (OU "deplacer et paralleliser")
    modules = "system LevierDeplacer, Deplacer, verificationPrecedence;"
else:
    if levier == "paralleliser":
        modules = "system LevierParallelisation, Masquer, verificationParallelisation,
Deplacer, verificationPrecedence;"
    else:
        if levier == "déplacer et paralléliser"
            modules = "system LevierDeplacer, Deplacer, verificationPrecedence,
LevierParallelisation, Masquer, verificationParallelisation;"
        else:
            if levier == "chevaucher":
                modules = ...

# Ouverture du fichier source
with open(template, 'r') as f:
    sourceLines = f.readlines()

with open(model_file, "w") as destination:
    destination.write("<system>\n")
    destination.write("%s\n" %(modules))
    destination.write("</system>\n")
```

I.3. Appel du logiciel de résolutions

En python, on appelle un fichier batch qui lance alors le solveur

Python :

```
os.system(".\SolverLaunch.bat " + model_file + " " + properties_file + " " + trace_file)
```

Fichier batch :

```
REM %1 est le modèle
REM %2 le fichier de propriétés
REM %3 le fichier trace enregistré
REM -o0 Breadth first.
REM -o1 Depth first.
REM -o2 Random depth first.
REM -t0 pour obtenir une trace
verifysa.exe -o2 -t0 -y -T %1 %2 2>%3
pause
```

I.4. Analyse des résultats

L'analyse des résultats consiste à récupérer les valeurs du dernier état de la trace dont un exemple est donné ci-dessous. Nous pouvons ainsi récupérer le nombre de déplacement, le nombre de parallélisation, la durée de chaque poste.

```
State
(LevierParallelisation._id13  Masquer._id17  verificationParallelisation._id19  Deplacer._id6
verificationPrecedence._id8 )
listeDesDurees[0]=0
[Liste des valeurs]
    listeAllocation[0]=0 listeAllocation[1]=0 listeAllocation[2]=0 listeAllocation[3]=-1

Transitions:
LevierParallelisation._id13->LevierParallelisation._id12 { 27 == 28 + 1 || 27 + 1 == 28, tau,
tacheAdeplacer := 27, operationCible := listeAllocation[28], tacheAdeplacer2 := 28 }

State
( LevierParallelisation._id12  Masquer._id17  verificationParallelisation._id19  Deplacer._id6
verificationPrecedence._id8 )
[Liste des valeurs]
    listeAllocation[0]=0 listeAllocation[1]=0 listeAllocation[2]=0 listeAllocation[3]=-1
```

II. La propriété

```
E<> (forall (j:id_O) ( (sum (i:id_T) ( listeDesDurees[i] * (j == listeAllocation[i]) )) <= takt + epsilon))
```

III. Déclaration globale

```
// Place global declarations here.
int DureeVerif[8]= {0,0,0,0,0,0,0,0};

int NbChangements = 0;
int NbParallelisation = 0;

// Tâches
const int takt = 4462;
const int epsilon = 100;
```

Annexe 2 : Implémentation de la méthode proposée

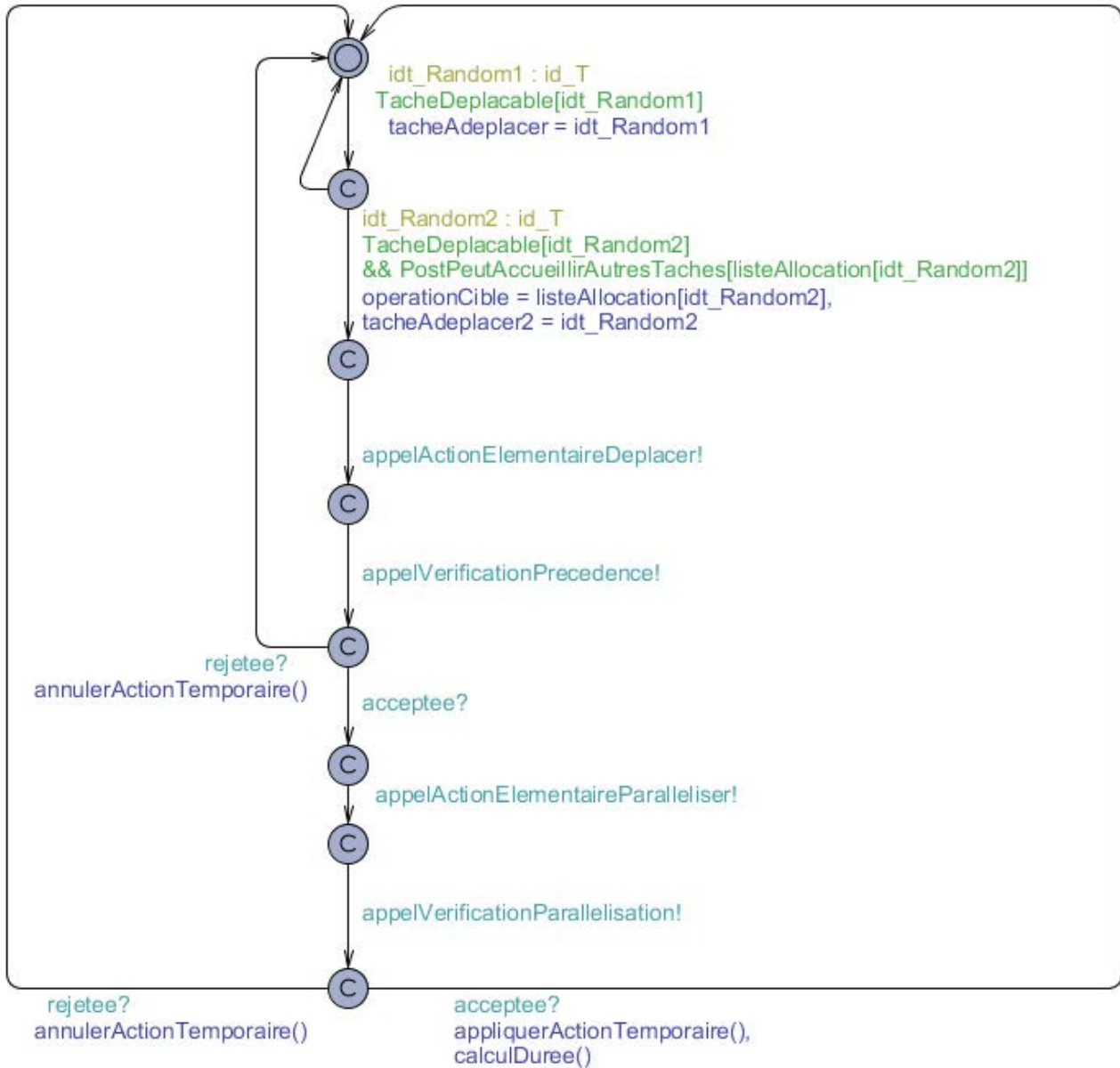
[illegible]

[illegible]

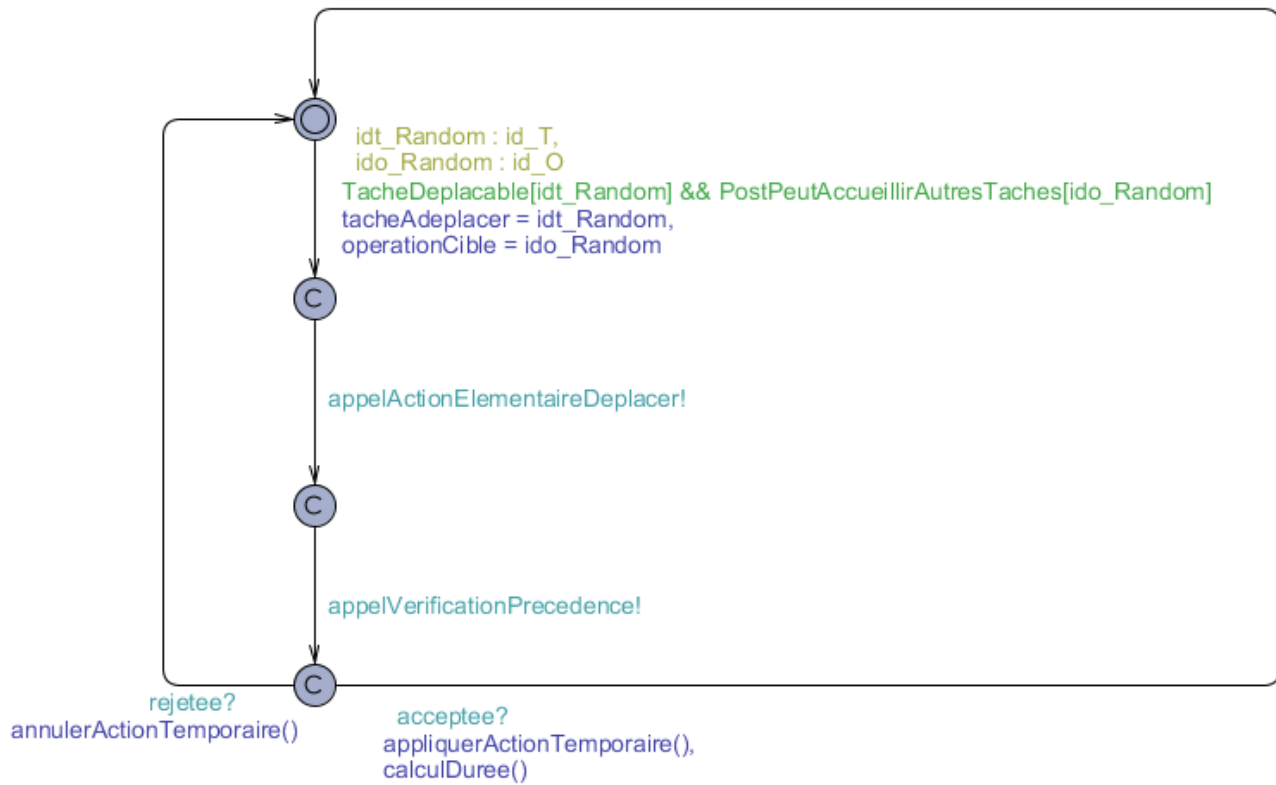
IV. Les modules

IV.1. Modules maître

IV.1.1. Levier « paralléliser »

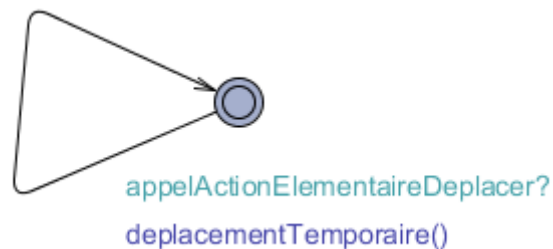


IV.4.1.2. Levier « déplacer »



IV.2. Modules d'action

IV.2.1. Action « déplacer »

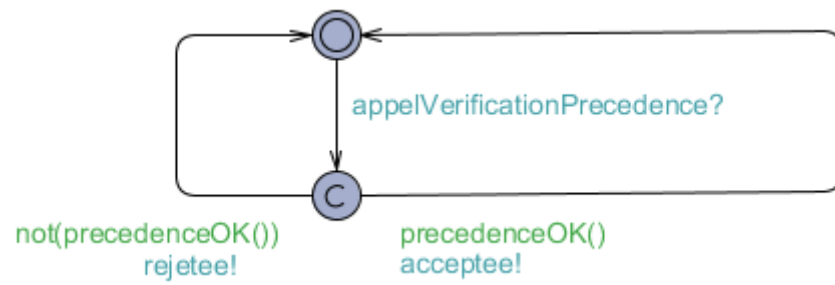


IV.2.2. Action « paralléliser »

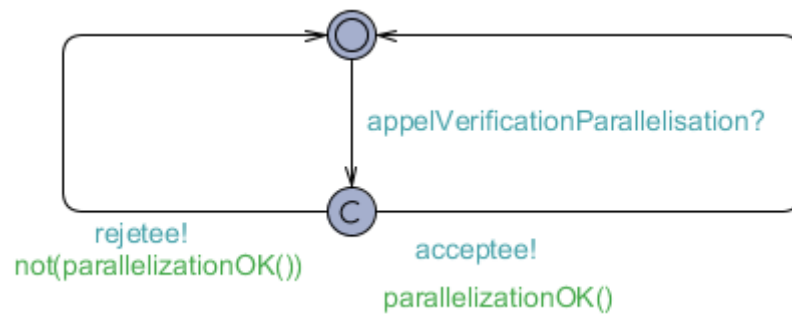


IV.3. Modules de vérification

IV.3.1. Vérification de la précedence



IV.3.2. Vérification de la parallélisation



Annexe 3 : Applications développées pour l'entreprise Trane

I. Suivi du takt time

I.1. Évolution au cours du temps, saisie manuelle

Date	Equipe	Somme min	nb Op	prod	Op 1/2	TSF	OP 4/5	Rob	Auto	Tub	Dudg
21/09/11	A-Midi	4966	17	292	223	160	265	286	400	505	194
22/09/11	matin	4242	17,8	238	222	98	582	94	87	391	397
22/09/11	A-Midi	5497	17,5	314	471	290	383	230	331	538	318
23/09/11	matin	4104	16,8	244	319	96	596	100		433	299
23/09/11	vendredi A-Midi	3890	17,5	333	515	166	299	240	244	297	188
26/09/11	matin	4899	17,9	274	231	254	281	236	399	300	231
28/09/11	matin	6042	18,0	336	307	300	576	244	382	277	271
28/09/11	A-Midi	4309	15,5	278	278	164	478	196	345	208	219
29/09/11	matin	4112	19	216	153	242	283	192	310	120	115
30/09/11	matin	5257,9	17	309	398	238	403	200	296	277	357
03/10/11	matin	5604	17,0	330	448	238	403	240	296	314	357
03/10/11	A-Midi	4615	18	256	374	80	254	192	272	267	420
04/10/11	matin	5518	17,0	325	240	360	416	234	307	237	473

I.2. Visualisation en temps réel

Suivi des backflush Epinal Operations

Lieu	Ligne	Nombre de backflush	Objectif de l'équipe	Nombre de backflush / Objectif du jour
CHA	AC	0	6	0 / 12
CHA	WC	0	3	0 / 5
GOL	SCROLL	1	12	2 / 12
GOL	V1	0	0	0 / 0
GOL	V2-V3	0	3	0 / 6

Si le takt est respecté, l'affichage est vert. Si le produit n'est pas sorti avant le takt time, il y a un retard orange si moins de 30 minutes, rouge si plus.

II. Suivi des en-cours

Cette application a pour but de connaître en temps réel les en-cours présents dans la ligne, mais aussi de calculer en temps réel le respect du takt time.

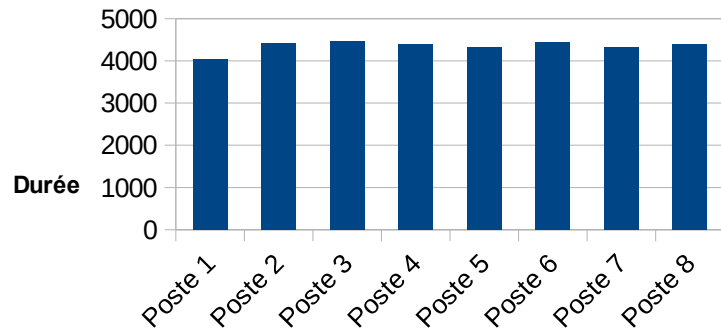
Cette application est basée sur un ensemble de scanners de codes-barre en entrée et en sortie de certaines portions de la ligne. Le nombre de produits sortant de la ligne est alors connu. Ce score est comparé en temps réel à l'objectif et mis en parallèle du takt attendu afin d'afficher si la production est en retard ou si elle est dans les temps. Chaque événement étant enregistré et horodaté, une analyse peut ensuite être effectuée.

The screenshot shows a Windows-style application window titled "Suivi des en-cours". The window has a menu bar with the following options: "Déconnection", "Modifier l'identifiant du poste", "Modifier les objectifs", "Modifier affichage au démarrage", and "Affichage". Below the menu bar, the text "LS DEBUT Sortie" is displayed. There is a large empty rectangular box for scanning, preceded by the text "OF scanné :". Below this box, the text "Dernier scan :" and "Heure dernier scan :" is visible. A red banner at the bottom of the window displays the large white numbers "01 / 07". Above these numbers, in smaller white text, it says "Heure dernière mise à jour : 22/10/2014 09:12:58". Below the red banner, the word "Réalisé" is on the left and "Objectif journée" is on the right.

Annexe 4 : Données du troisième cas d'étude

Équilibrage initial du troisième cas d'étude.

Identifiant Poste	Durée
Poste 1	4045
Poste 2	4432
Poste 3	4462
Poste 4	4393
Poste 5	4317
Poste 6	4450
Poste 7	4336
Poste 8	4403



Identifiant tâche	Durée	Allocation initiale	Identifiant tâche	Durée	Allocation initiale	Identifiant tâche	Durée	Allocation initiale
1	10	1	63	60	3	125	105	4
2	90	1	64	120	3	126	60	4
3	30	1	65	30	3	127	105	4
4	270	1	66	187	3	128	150	4
5	45	1	67	120	3	129	105	4
6	30	1	68	75	3	130	60	4
7	60	1	69	90	3	131	30	4
8	30	1	70	60	3	132	30	4
9	90	1	71	60	4	133	60	4
10	30	1	72	120	4	134	45	4
11	45	1	73	60	4	135	30	4
12	45	1	74	180	4	136	60	4
13	30	1	75	30	4	137	30	4
14	30	1	76	30	4	138	60	4
15	30	1	77	45	4	139	105	4
16	1380	1	78	15	4	140	60	4
17	60	1	79	30	4	141	75	4
18	150	1	80	30	4	142	30	4
19	30	1	81	24	4	143	120	4
20	15	1	82	30	4	144	60	4
21	30	1	83	15	4	145	45	4
22	270	1	84	12	4	146	60	4
23	30	1	85	15	4	147	30	5
24	180	1	86	15	4	148	30	5
25	105	1	87	30	4	149	18	5
26	30	1	88	12	4	150	45	5
27	120	1	89	15	4	151	60	5
28	105	1	90	30	4	152	90	5
29	480	1	91	60	4	153	15	5
30	105	1	92	15	4	154	60	5
31	90	1	93	15	4	155	30	5
32	1380	2	94	45	4	156	90	5
33	150	2	95	60	4	157	30	5
34	50	2	96	12	4	158	30	5
35	60	2	97	15	4	159	30	5
36	60	2	98	15	4	160	45	5
37	120	2	99	45	4	161	45	5
38	525	2	100	30	4	162	60	5
39	120	2	101	24	4	163	180	5
40	60	2	102	30	4	164	45	5
41	60	2	103	30	4	165	45	5
42	30	2	104	24	4	166	60	5
43	60	2	105	30	4	167	60	5
44	60	2	106	15	4	168	60	5
45	120	2	107	30	4	169	180	5
46	60	2	108	12	4	170	60	5
47	120	2	109	60	4	171	30	5
48	60	2	110	30	4	172	60	5
49	30	2	111	105	4	173	75	5
50	437	2	112	105	4	174	60	5
51	105	2	113	374	4	175	30	5
52	15	2	114	105	4	176	150	5
53	60	2	115	45	4	177	60	5
54	150	2	116	374	4	178	60	5
55	45	2	117	60	4	179	15	5
56	60	2	118	60	4	180	60	5
57	240	2	119	45	4	181	30	5
58	60	2	120	30	4	182	30	5
59	60	2	121	30	4	183	30	5
60	75	2	122	45	4	184	420	5
61	3660	3	123	90	4	185	30	5
62	60	3	124	45	4	186	120	5

Annexe 4 : Données du troisième cas d'étude

Identifiant tâche	Durée	Allocation initiale	Identifiant tâche	Durée	Allocation initiale
187	120	5	249	45	7
188	525	5	250	90	7
189	45	5	251	135	7
190	15	5	252	30	7
191	45	5	253	15	7
192	45	5	254	840	7
193	60	5	255	150	7
194	150	5	256	60	7
195	150	5	257	30	7
196	75	5	258	100	8
197	60	5	259	156	8
198	60	5	260	187	8
199	30	5	261	45	8
200	120	5	262	180	8
201	30	5	263	105	8
202	75	5	264	30	8
203	30	5	265	75	8
204	24	5	266	30	8
205	90	6	267	150	8
206	3660	6	268	90	8
207	15	6	269	90	8
208	15	6	270	360	8
209	60	6	271	75	8
210	40	6	272	90	8
211	480	6	273	30	8
212	90	6	274	30	8
213	15	7	275	30	8
214	45	7	276	15	8
215	40	7	277	210	8
216	60	7	278	210	8
217	45	7	279	120	8
218	15	7	280	60	8
219	45	7	281	45	8
220	156	7	282	60	8
221	30	7	283	75	8
222	30	7	284	45	8
223	30	7	285	30	8
224	75	7	286	150	8
225	60	7	287	60	8
226	60	7	288	30	8
227	60	7	289	30	8
228	30	7	290	120	8
229	150	7	291	75	8
230	60	7	292	30	8
231	30	7	293	120	8
232	120	7	294	30	8
233	105	7	295	45	8
234	15	7	296	45	8
235	390	7	297	30	8
236	390	7	298	60	8
237	60	7	299	180	8
238	15	7	300	60	8
239	15	7	301	45	8
240	60	7	302	60	8
241	105	7	303	30	8
242	75	7	304	60	8
243	105	7	305	180	8
244	75	7	306	60	8
245	180	7	307	30	8
246	30	7	308	90	8
247	90	7	309	60	8
248	75	7			

Annexe 5 : Résultats du troisième cas d'étude

Retard	champ d'application	levier	exploration	Durée de résolution	Nb déplacements	Nb parallélisation	Nombre d essais inabouti	durée résolution (=durée +essais * 10s)	Nb changements (=déplacements + paralléli
t064	ligne	déplacer	breadth first	3,962	1,000	0,000	0,000	3,962	1,000
t064	ligne	déplacer	breadth first	3,900	1,000	0,000	0,000	3,900	1,000
t064	ligne	déplacer	breadth first	24,632	1,000	0,000	0,000	24,632	1,000
t064	ligne	déplacer	breadth first	24,148	1,000	0,000	0,000	24,148	1,000
t064	ligne	déplacer	breadth first	24,290	1,000	0,000	0,000	24,290	1,000
t064	ligne	déplacer	random depth	3,441	2,000	0,000	0,000	3,441	2,000
t064	ligne	déplacer	random depth	3,224	1,000	0,000	4,000	43,224	1,000
t064	ligne	déplacer	random depth	3,343	1,000	0,000	16,000	163,343	1,000
t064	ligne	déplacer	random depth	6,583	13,000	0,000	12,000	126,583	13,000
t064	ligne	déplacer	random depth	3,931	1,000	0,000	3,000	33,931	1,000
t064	ligne	déplacer et paralléliser	breadth first	116,782	1	0	0	116,782	1,000
t064	ligne	déplacer et paralléliser	breadth first	113,382	1	0	0	113,382	1,000
t064	ligne	déplacer et paralléliser	breadth first	117,983	1	0	0	117,983	1,000
t064	ligne	déplacer et paralléliser	breadth first	116,313	1	0	0	116,313	1,000
t064	ligne	déplacer et paralléliser	breadth first	121,353	1	0	0	121,353	1,000
t064	ligne	déplacer et paralléliser	random depth	7,223	7	1	3	37,223	8,000
t064	ligne	déplacer et paralléliser	random depth	6,521	2	1	4	46,521	3,000
t064	ligne	déplacer et paralléliser	random deth	7,886	6	0	2	27,886	6,000
t064	ligne	déplacer et paralléliser	random deth	7,232	3	1	7	77,232	4,000
t064	ligne	déplacer et paralléliser	random deth	6,003	1	0	16	166,003	1,000
t064	ligne	paralléliser	breadth first	69,657	0	1	0	69,657	1,000
t064	ligne	paralléliser	breadth first	62,618	0	1	0	62,618	1,000
t064	ligne	paralléliser	breadth first	60,31	0	1	0	60,310	1,000
t064	ligne	paralléliser	breadth first	62,229	0	1	0	62,229	1,000
t064	ligne	paralléliser	breadth first	63,149	0	1	0	63,149	1,000
t064	ligne	paralléliser	random depth	3,273	0	9	7	73,273	9,000
t064	ligne	paralléliser	random depth	7,697	0	114	0	7,697	114,000
t064	ligne	paralléliser	random depth	4,661	0	30	0	4,661	30,000
t064	ligne	paralléliser	random depth	3,622	0	15	0	3,622	15,000
t064	ligne	paralléliser	random depth	7,024	0	101	1	17,024	101,000
t064	restreint	déplacer	breadth first	3,432	1,000	0,000	0	3,432	1,000
t064	restreint	déplacer	breadth first	3,385	1,000	0,000	0	3,385	1,000
t064	restreint	déplacer	breadth first	3,261	1,000	0,000	0	3,261	1,000
t064	restreint	déplacer	breadth first	3,417	1,000	0,000	0	3,417	1,000
t064	restreint	déplacer	breadth first	3,37	1,000	0,000	0	3,370	1,000
t064	restreint	déplacer	random depth	200	1	0	0	200,000	1,000
t064	restreint	déplacer	random depth	200	1	0	0	200,000	1,000
t064	restreint	déplacer	random depth	200	1	0	0	200,000	1,000
t064	restreint	déplacer	random depth	200	1	0	0	200,000	1,000
t064	restreint	déplacer	random depth	200	1	0	0	200,000	1,000
t064	restreint	déplacer et paralléliser	breadth first	14,740	1,000	0,000	0	14,740	1,000
t064	restreint	déplacer et paralléliser	breadth first	13,787	1,000	0,000	0	13,787	1,000
t064	restreint	déplacer et paralléliser	breadth first	13,471	1,000	0,000	0	13,471	1,000
t064	restreint	déplacer et paralléliser	breadth first	3,998	1,000	0,000	0	3,998	1,000
t064	restreint	déplacer et paralléliser	breadth first	13,491	1,000	0,000	0	13,491	1,000
t064	restreint	déplacer et paralléliser	random depth	6,112	2,000	0,000	0,000	6,112	2,000
t064	restreint	déplacer et paralléliser	random depth	5,416	2,000	0,000	0,000	5,416	2,000
t064	restreint	déplacer et paralléliser	random depth	5,913	2,000	4,000	5,000	55,913	6,000
t064	restreint	déplacer et paralléliser	random depth	5,928	1,000	5,000	1,000	15,928	6,000
t064	restreint	déplacer et paralléliser	random depth	5,850	4,000	2,000	6,000	65,850	6,000
t064	restreint	paralléliser	breadth first	11,887	0,000	1,000	0	11,887	1,000
t064	restreint	paralléliser	breadth first	12,296	0,000	1,000	0	12,296	1,000
t064	restreint	paralléliser	breadth first	12,028	0,000	1,000	0	12,028	1,000
t064	restreint	paralléliser	breadth first	11,794	0,000	1,000	0	11,794	1,000
t064	restreint	paralléliser	breadth first	11,949	0,000	1,000	0	11,949	1,000
t064	restreint	paralléliser	random depth	2,534	0,000	5,000	0,000	2,534	5,000
t064	restreint	paralléliser	random depth	2,974	0,000	1,000	1,000	12,974	1,000
t064	restreint	paralléliser	random depth	2,597	0,000	1,000	0,000	2,597	1,000
t064	restreint	paralléliser	random depth	2,481	0,000	1,000	0,000	2,481	1,000
t064	restreint	paralléliser	random depth	2,636	0,000	9,000	0,000	2,636	9,000

Annexe 5 : Résultats du troisième cas d'étude

Retard	champ d'application	levier	exploration	Durée de résolution	Nb déplacements	Nb parallélisation	Nombre d essais inabouti	durée résolution (=durée +essais * 10s)	Nb changements (=déplacements +paralléli
t150	ligne	déplacer	breadth first	3,876	1,000	0,000	0,000	3,876	1,000
t150	ligne	déplacer	breadth first	5,047	1,000	0,000	0,000	5,047	1,000
t150	ligne	déplacer	breadth first	5,107	1,000	0,000	0,000	5,107	1,000
t150	ligne	déplacer	breadth first	5,104	1,000	0,000	0,000	5,104	1,000
t150	ligne	déplacer	breadth first	5,033	1,000	0,000	0,000	5,033	1,000
t150	ligne	déplacer	random depth	3,105	1,000	0,000	2,000	23,105	1,000
t150	ligne	déplacer	random depth	3,104	1,000	0,000	0,000	3,104	1,000
t150	ligne	déplacer	random depth	3,791	2,000	0,000	0,000	3,791	2,000
t150	ligne	déplacer	random depth	3,682	3,000	0,000	0,000	3,682	3,000
t150	ligne	déplacer	random depth	4,680	3,000	0,000	0,000	4,680	3,000
t150	ligne	déplacer et paralléliser	breadth first	69,252	1,000	0,000	0,000	69,252	1,000
t150	ligne	déplacer et paralléliser	breadth first	62,352	1,000	0,000	0,000	62,352	1,000
t150	ligne	déplacer et paralléliser	breadth first	75,426	1,000	0,000	0,000	75,426	1,000
t150	ligne	déplacer et paralléliser	breadth first	88,231	1,000	0,000	0,000	88,231	1,000
t150	ligne	déplacer et paralléliser	breadth first	62,186	1,000	0,000	0,000	62,186	1,000
t150	ligne	déplacer et paralléliser	random depth	5,788	2,000	0,000	1,000	15,788	2,000
t150	ligne	déplacer et paralléliser	random depth	5,679	1,000	0,000	0,000	5,679	1,000
t150	ligne	déplacer et paralléliser	random depth	6,318	2,000	2,000	1,000	16,318	4,000
t150	ligne	déplacer et paralléliser	random depth	5,663	0,000	1,000	0,000	5,663	1,000
t150	ligne	déplacer et paralléliser	random depth	5,897	1,000	0,000	0,000	5,897	1,000
t150	ligne	paralléliser	breadth first	56,227	0,000	1,000	0,000	56,227	1,000
t150	ligne	paralléliser	breadth first	57,790	0,000	1,000	0,000	57,790	1,000
t150	ligne	paralléliser	breadth first	41,917	0,000	1,000	0,000	41,917	1,000
t150	ligne	paralléliser	breadth first	41,762	0,000	1,000	0,000	41,762	1,000
t150	ligne	paralléliser	breadth first	41,543	0,000	1,000	0,000	41,543	1,000
t150	ligne	paralléliser	random depth	2,746	0,000	4,000	0,000	2,746	4,000
t150	ligne	paralléliser	random depth	2,730	0,000	3,000	0,000	2,730	3,000
t150	ligne	paralléliser	random depth	2,605	0,000	2,000	0,000	2,605	2,000
t150	ligne	paralléliser	random depth	2,574	0,000	1,000	0,000	2,574	1,000
t150	ligne	paralléliser	random depth	2,652	0,000	2,000	0,000	2,652	2,000
t150	restreint	déplacer	breadth first	3,338	1,000	0,000	0,000	3,338	1,000
t150	restreint	déplacer	breadth first	3,260	1,000	0,000	0,000	3,260	1,000
t150	restreint	déplacer	breadth first	3,276	1,000	0,000	0,000	3,276	1,000
t150	restreint	déplacer	breadth first	3,245	1,000	0,000	0,000	3,245	1,000
t150	restreint	déplacer	breadth first	3,213	1,000	0,000	0,000	3,213	1,000
t150	restreint	déplacer	random depth	3,152	1,000	0,000	0,000	3,152	1,000
t150	restreint	déplacer	random depth	3,026	1,000	0,000	0,000	3,026	1,000
t150	restreint	déplacer	random depth	3,152	1,000	0,000	1,000	13,152	1,000
t150	restreint	déplacer	random depth	3,541	5,000	0,000	0,000	3,541	5,000
t150	restreint	déplacer	random depth	3,058	1,000	0,000	0,000	3,058	1,000
t150	restreint	déplacer et paralléliser	breadth first	25,818	1,000	0,000	0,000	25,818	1,000
t150	restreint	déplacer et paralléliser	breadth first	27,238	1,000	0,000	0,000	27,238	1,000
t150	restreint	déplacer et paralléliser	breadth first	26,598	1,000	0,000	0,000	26,598	1,000
t150	restreint	déplacer et paralléliser	breadth first	26,941	1,000	0,000	0,000	26,941	1,000
t150	restreint	déplacer et paralléliser	breadth first	24,726	1,000	0,000	0,000	24,726	1,000
t150	restreint	déplacer et paralléliser	random depth	5,382	0,000	1,000	0,000	5,382	1,000
t150	restreint	déplacer et paralléliser	random depth	5,585	0,000	2	0,000	5,585	2,000
t150	restreint	déplacer et paralléliser	random depth	5,491	0,000	1,000	0,000	5,491	1,000
t150	restreint	déplacer et paralléliser	random depth	5,428	1,000	0,000	0,000	5,428	1,000
t150	restreint	déplacer et paralléliser	random depth	8,845	19,000	19,000	0,000	8,845	38,000
t150	restreint	paralléliser	breadth first	17,472	0,000	1,000	0,000	17,472	1,000
t150	restreint	paralléliser	breadth first	17,441	0,000	1,000	0,000	17,441	1,000
t150	restreint	paralléliser	breadth first	17,394	0,000	1,000	0,000	17,394	1,000
t150	restreint	paralléliser	breadth first	17,519	0,000	1,000	0,000	17,519	1,000
t150	restreint	paralléliser	breadth first	17,254	0,000	1,000	0,000	17,254	1,000
t150	restreint	paralléliser	random depth	2,574	0,000	1,000	0,000	2,574	1,000
t150	restreint	paralléliser	random depth	2,512	0,000	1,000	0,000	2,512	1,000
t150	restreint	paralléliser	random depth	2,527	0,000	1,000	0,000	2,527	1,000
t150	restreint	paralléliser	random depth	2,684	0,000	2,000	0,000	2,684	2,000
t150	restreint	paralléliser	random depth	2,683	0,000	1,000	0,000	2,683	1,000

Retard	champ d'application	levier	exploration	Durée de résolution	Nb déplacements	Nb parallélisation	Nombre d essais inaboutis	durée résolution (=durée +essais * 10s)	Nb changements (=déplacements +parallélisation)
t250	ligne	déplacer	breadth first	3,354	1,000	0,000	0,000	3,354	1,000
t250	ligne	déplacer	breadth first	3,291	1,000	0,000	0,000	3,291	1,000
t250	ligne	déplacer	breadth first	3,167	1,000	0,000	0,000	3,167	1,000
t250	ligne	déplacer	breadth first	3,213	1,000	0,000	0,000	3,213	1,000
t250	ligne	déplacer	breadth first	3,338	1,000	0,000	0,000	3,338	1,000
t250	ligne	déplacer	random depth	3,573	1,000	0,000	2,000	23,573	1,000
t250	ligne	déplacer	random depth	3,042	1,000	0,000	9,000	93,042	1,000
t250	ligne	déplacer	random depth	3,759	10,000	0,000	2,000	23,759	10,000
t250	ligne	déplacer	random depth	3,104	1,000	0,000	0,000	3,104	1,000
t250	ligne	déplacer	random depth	4,087	19,000	0,000	4,000	44,087	19,000
t250	ligne	déplacer et paralléliser	breadth first	13,994	1,000	0,000	0,000	13,994	1,000
t250	ligne	déplacer et paralléliser	breadth first	14,477	1,000	0,000	0,000	14,477	1,000
t250	ligne	déplacer et paralléliser	breadth first	14,851	1,000	0,000	0,000	14,851	1,000
t250	ligne	déplacer et paralléliser	breadth first	76,175	1,000	0,000	0,000	76,175	1,000
t250	ligne	déplacer et paralléliser	breadth first	75,160	1,000	0,000	0,000	75,160	1,000
t250	ligne	déplacer et paralléliser	random depth	5,632	1,000	1,000	0,000	5,632	2,000
t250	ligne	déplacer et paralléliser	random depth	5,320	0,000	1,000	2,000	25,320	1,000
t250	ligne	déplacer et paralléliser	random depth	6,771	13,000	8,000	0,000	6,771	21,000
t250	ligne	déplacer et paralléliser	random depth	5,304	1,000	0,000	1,000	15,304	1,000
t250	ligne	déplacer et paralléliser	random depth	6,599	16,000	10,000	0,000	6,599	26,000
t250	ligne	paralléliser	breadth first	9,687	0,000	1,000	0,000	9,687	1,000
t250	ligne	paralléliser	breadth first	9,547	0,000	1,000	0,000	9,547	1,000
t250	ligne	paralléliser	breadth first	9,547	0,000	1,000	0,000	9,547	1,000
t250	ligne	paralléliser	breadth first	9,547	0,000	1,000	0,000	9,547	1,000
t250	ligne	paralléliser	breadth first	9,922	0,000	1,000	0,000	9,922	1,000
t250	ligne	paralléliser	random depth	2,621	0,000	1,000	0,000	2,621	1,000
t250	ligne	paralléliser	random depth	2,698	0,000	5,000	0,000	2,698	5,000
t250	ligne	paralléliser	random depth	3,152	0,000	23,000	0,000	3,152	23,000
t250	ligne	paralléliser	random depth	2,715	0,000	4,000	0,000	2,715	4,000
t250	ligne	paralléliser	random depth	2,636	0,000	3,000	0,000	2,636	3,000
t250	restreint	déplacer	breadth first	3,776	1,000	0,000	0,000	3,776	1,000
t250	restreint	déplacer	breadth first	3,322	1,000	0,000	0,000	3,322	1,000
t250	restreint	déplacer	breadth first	3,307	1,000	0,000	0,000	3,307	1,000
t250	restreint	déplacer	breadth first	3,603	1,000	0,000	0,000	3,603	1,000
t250	restreint	déplacer	breadth first	3,307	1,000	0,000	0,000	3,307	1,000
t250	restreint	déplacer	random depth	3,073	1,000	0,000	3,000	33,073	1,000
t250	restreint	déplacer	random depth	3,073	1,000	0,000	5,000	53,073	1,000
t250	restreint	déplacer	random depth	3,198	3,000	0,000	0,000	3,198	3,000
t250	restreint	déplacer	random depth	3,760	15,000	0,000	0,000	3,760	15,000
t250	restreint	déplacer	random depth	3,385	9,000	0,000	0,000	3,385	9,000
t250	restreint	déplacer et paralléliser	breadth first	77,298	1,000	0,000	0,000	77,298	1,000
t250	restreint	déplacer et paralléliser	breadth first	76,050	1,000	0,000	0,000	76,050	1,000
t250	restreint	déplacer et paralléliser	breadth first	82,053	1,000	0,000	0,000	82,053	1,000
t250	restreint	déplacer et paralléliser	breadth first	77,391	1,000	0,000	0,000	77,391	1,000
t250	restreint	déplacer et paralléliser	breadth first	75,754	1,000	0,000	0,000	75,754	1,000
t250	restreint	déplacer et paralléliser	random depth	7,020	14,000	17,000	0,000	7,020	31,000
t250	restreint	déplacer et paralléliser	random depth	5,273	1,000	0,000	1,000	15,273	1,000
t250	restreint	déplacer et paralléliser	random depth	5,882	3,000	2,000	0,000	5,882	5,000
t250	restreint	déplacer et paralléliser	random depth	5,647	1,000	3,000	3,000	35,647	4,000
t250	restreint	déplacer et paralléliser	random depth	6,271	5,000	9,000	0,000	6,271	14,000
t250	restreint	paralléliser	breadth first	10,015	0,000	1,000	0,000	10,015	1,000
t250	restreint	paralléliser	breadth first	9,610	0,000	1,000	0,000	9,610	1,000
t250	restreint	paralléliser	breadth first	9,563	0,000	1,000	0,000	9,563	1,000
t250	restreint	paralléliser	breadth first	10,062	0,000	1,000	0,000	10,062	1,000
t250	restreint	paralléliser	breadth first	9,625	0,000	1,000	0,000	9,625	1,000
t250	restreint	paralléliser	random depth	2,589	0,000	2,000	0,000	2,589	2,000
t250	restreint	paralléliser	random depth	3,011	0,000	15,000	0,000	3,011	15,000
t250	restreint	paralléliser	random depth	2,621	0,000	3,000	0,000	2,621	3,000
t250	restreint	paralléliser	random depth	3,198	0,000	27,000	0,000	3,198	27,000
t250	restreint	paralléliser	random depth	2,621	0,000	1,000	1,000	12,621	1,000

Résumé : L'organisation de la production en ligne d'assemblage a fait ses preuves depuis le début de l'industrialisation et reste d'actualité malgré le passage de la production de masse à une production de plus en plus personnalisée et fluctuante. Pour s'assurer de bien répondre à la demande client (quantité, délai, etc.) et de réduire les coûts de fabrication, les lignes doivent être équilibrées de telle manière que les produits passent d'un poste à un autre selon une même cadence, appelée takt time. Si le problème de conception de ces lignes est largement étudié dans la littérature sous le nom d'ALBP (Assembly Line Balancing Problems), très peu de travaux concernent leur reconfiguration (rééquilibrage) et en particulier la reconfiguration dans un horizon court terme et en-ligne que nous appellerons le « rééquilibrage dynamique ». Les travaux présentés dans ce mémoire portent donc sur le rééquilibrage dynamique des lignes d'assemblage en réaction aux événements court terme perturbants la production. L'objectif étant de changer l'allocation des tâches à réaliser sur les postes de travail afin de lisser la charge induite par la perturbation. Cela consiste donc à trouver une solution (pas nécessairement optimale) en ligne (sans arrêts de production et dans un délai très court). Nous nous sommes intéressés dans un premier temps aux méthodes de résolution approchées pour leur temps de résolution rapide. Deux approches ont été étudiées pour leur simplicité, la rapidité de leur mise en œuvre et leur nouveauté dans le domaine du ALBP : la méthode ILS (Iterated Local Search) et la recherche d'atteignabilité sur un réseau d'automates communicants. La conclusion de cette étude nous a montré que les deux approches peuvent répondre au besoin de réactivité imposé par notre problème mais qu'elles sont difficilement adaptables lorsque les leviers d'action pour rééquilibrer la ligne sont modifiés. Notre proposition pour répondre à ce besoin d'adaptabilité et de généricité est une méthodologie de modélisation et de résolution des problèmes de rééquilibrage des lignes d'assemblage s'appuyant sur un réseau d'automates communicants et une résolution par une recherche d'atteignabilité sur celui-ci. Pour s'assurer de la généricité de la méthodologie, celle-ci est décomposée en deux niveaux d'abstraction. Le premier concerne la classe du problème ALBP considéré et le second est centré sur le problème spécifique à résoudre. Les modèles ILS et SED ainsi que notre approche ont été validés sur des exemples industriels issus de la société Trane.

Abstract : Assembly lines are flow-oriented production systems. They are still typical in industrial production systems despite the evolution from mass-production to more personalized and fluctuating production. In order to meet customer demand (quantity, time, etc.) and to reduce manufacturing costs, the lines must be well balanced. In other words, the products need to move from one workstation to another with the same pace, usually called takt time. This assembly line design problem is widely studied in the literature under the name of ALBP (Assembly Line Balancing Problems). Nonetheless, very few studies concern their reconfiguration (or rebalancing) in particular on-line rebalancing faced in a short-term horizon we called this reconfiguration dynamic rebalancing. The work presented in this thesis focuses on the dynamic rebalancing of assembly lines in response to short-term events disturbing production. Two approaches have been studied for their simplicity, speed of implementation and innovation in the field of ALBP: the ILS approach (Iterated Local Search) and the reachability analysis approach based on a network of communicating automata. The conclusion of this study showed that both approaches can address the need for a quick resolution required by dynamic rebalancing problem but lacked for adaptation when model constraints or objectives are changed. A methodological approach for assembly line balancing problems modeling and resolution is proposed to bridge the gap of adaptability and genericity. The methodology is composed of two levels of abstraction to ensure the genericity of obtained models. The first deals with a study of the class of the considered problem. The second focuses on the specific problem to solve. ILS approach, SED models and the methodology have been validated on industrial examples from Trane factories.