Compromis entre robustesse et coût pour la configuration d'un parc de machines parallèles partiellement multifonctions

Alexis Aubry, Marie-Laure Espinouse, Mireille Jacomino, Pawel Mrozick

Compromis entre robustesse et coût pour la configuration d'un parc de machines parallèles partiellement multifonctions

Alexis Aubry, Marie-Laure Espinouse, Mireille Jacomino, Pawel Mrozicki
Laboratoire d'Automatique de Grenoble
Ensieg - BP46
38402 Saint-Martin D'Hères Cedex
{prenom.nom}@lag.ensieg.inpg.fr

Résumé : Le problème industriel auquel nous nous intéressons concerne la configuration d'un atelier particulier de la chaîne de fabrication de circuits intégrés. Configurer ce parc de machines revient à choisir quelles machines peuvent traiter quels types de produit en les munissant des ressources adéquates et en faisant les réglages appropriés. La mise en place des ressources et leur maintenance constituent le coût de la configuration. L'objectif est de minimiser ce coût tout en garantissant un délai de fabrication.
Par ailleurs, les machines ayant un coût particulièrement élevé, l'industriel souhaite que la demande soit satisfaite en équilibrant la charge de travail sur les différentes machines. Dans le secteur d'activité considéré, la demande est extrêmement variable dans le temps. La configuration doit cependant garantir les performances d'équilibre de la charge malgré les incertitudes : elle doit être robuste.
Après avoir modélisé le problème, une méthode basée sur des outils de l'optimisation combinatoire a été proposée et implantée. Des résultats exacts sur des exemples académiques ont été obtenus. Sur des exemples de taille industrielle, des solutions approchées de bonne qualité ont été calculées. Une analyse qualitative des résultats a permis de montrer que notre approche permet de répondre au problème industriel posé.
Mots-clés : Modélisation et commande des systèmes discrets, PSE, Robustesse en ordonnancement, Incertitudes sur la demande, Multi-Purpose Machines.

1 INTRODUCTION

Les enjeux économiques dans une entreprise sont tels qu'ils interviennent à tous les niveaux. C'est au niveau le plus bas, dans les ateliers de production, que par exemple le choix de l'ordre de passage des pièces sur une machine intervient pour remplir différents objectifs. Pour répondre à ces objectifs, un outil privilégié est l'ordonnancement. La plupart des travaux en ordonnancement supposent que toutes les caractéristiques des tâches à ordonnancer sont connues. Une solution prévisionnelle au problème d'optimisation est calculée. En général, son exécution dans l'atelier de fabrication ne correspond pas exactement aux données prévues. Les performances de la solution réalisée sont alors inconnues a priori. L'ordonnancement robuste a pour objectif de proposer des solutions pour lesquelles les performances sont garanties pour un ensemble de perturbations qui pourraient survenir lors de la réalisation de ces solutions.
L'objet de notre travail, présenté dans cet article, qui est la minimisation des coûts liés à la configuration d'un parc de machines polyvalentes dans l'industrie des semi-conducteurs, se place dans le contexte de l'ordonnancement robuste. Le problème industriel auquel on s'intéresse concerne un atelier particulier de la chaîne de fabrication des circuits intégrés : l'atelier de photolithographie. Cet atelier est constitué de machines parallèles partiellement multifonctions, c'est à dire que chaque machine est capable de traiter un sous-ensemble de produits. Pour ce faire, la machine doit subir une mise au point spécifique. Cette mise au point se traduit par un réglage de la machine. Lorsque la mise au point est effectuée, on dit que la machine est qualifiée pour le type de produit correspondant. L'ensemble des qualifications constitue la configuration de l'atelier.
En début de mois, au moment où est élaborée la configuration du parc de machines, seule une demande moyenne pour chaque type de produit est connue. Dans le secteur d'activité considéré, la demande est extrêmement variable, les machines et leur configuration représentent des coûts d'investissement et de fonctionnement importants, il faut donc minimiser le nombre de qualifications tout en garantissant un taux maximal d'utilisation des machines malgré les incertitudes sur la demande.

2 DE L'ORDONNANCEMENT EN GÉNÉRAL

Les problèmes classiques d'ordonnancement considèrent que toutes les données du problème sont parfaitement connues et statiques. Cependant il est quasiment impossible de retrouver ce contexte certain dans la réalisation des solutions sur les processus réels. C'est pourquoi la prise en compte des perturbations est apparue comme nécessaire depuis quelques années car ces perturbations peuvent largement dégrader les résultats de l'ordonnancement classique. L'enjeu de l'ordonnancement avec prise en compte des incertitudes est de garantir un niveau de performance en présence de perturbations. Notre problème s'inscrit bien dans cette optique puisque le but est de trouver, à partir d'une configuration existante d'un parc de machines, une configuration qui sera robuste vis-

à-vis des incertitudes sur la demande tout en minimisant les différents coûts d'exploitation.
Dans le prochain paragraphe nous rappellerons ce qu'est l'ordonnancement dans son acceptation classique. Et dans le second paragraphe nous introduirons la notion de robustesse en ordonnancement.

2.1 De l'ordonnancement classique

« *L'ordonnancement est la distribution des ressources dans le temps pour réaliser un ensemble d'activités* »[Baker, 1974]

En ordonnancement classique, les données du problème (cf. [Esquirol, 1999]) sont parfaitement connues et considérées comme statiques.
La définition de ces données permet de modéliser le problème statique d'ordonnancement. Une solution à ce problème qui permettra d'optimiser la fonction objectif est appelée solution optimale. En pratique le but est de trouver un algorithme le plus rapide possible qui trouve une solution optimale. Une telle méthode est appelée exacte. On range ces algorithmes en trois catégories de complexité : 1) algorithme polynomial 2) algorithme pseudo-polynomial 3) algorithme exponentiel. Pour en savoir plus sur la complexité des problèmes en ordonnancement le lecteur peut se référer à [Pinedo, 2002]. Lorsque la taille et la complexité du problème ne permet pas d'utiliser une méthode exacte, on cherche un algorithme qui retournera une solution approchée. On appelle ces algorithmes, des heuristiques : le but est de trouver en un temps satisfaisant, une solution de bonne qualité.

2.2 De l'ordonnancement en contexte incertain : introduction à la robustesse

Le caractère optimal d'une solution calculée dans un contexte certain peut être largement remis en cause lors de l'exécution par un aléa ou une imprécision des données du modèle. C'est pourquoi l'ordonnancement avec prise en compte des incertitudes est nécessaire. Un outil privilégié pour la prise en charge des incertitudes en ordonnancement est la robustesse. Une définition simple peut être "*garantie de performances*". Cette garantie de performance est inhérente au critère de robustesse que l'on choisit. Le critère choisi doit donc être pertinent pour le problème considéré. Les auteurs de [Billaut, 2005, Kouvelis, 1997, Rossi, 2003] dressent, entre autres, un état des lieux des critères les plus utilisés dans la littérature.
[Rossi, 2003] donne une définition mathématique de la robustesse : « *Étant donné une fonction objectif σ, un critère de robustesse λ_k, et un niveau de performance à garantir C_k, on dit que la solution S est robuste sur l'ensemble de scénarios I si et seulement si $\lambda_k(S, \sigma, I) \leq C_k$* ».
Cette définition a l'avantage de mettre en avant que le caractère robuste d'une solution est relatif à un critère de robustesse λ_k, à un risque déterminé (représenté par I) et à un niveau de performance globale C_k donnés.

• *Analogie avec la notion de robustesse pour les systèmes continus*
Dans [Rossi, 2003], un tableau (cf. Tab. 1) permet de comparer les notions de robustesse en ordonnancement et en automatique des systèmes continus. Pour mieux appréhender la robustesse des systèmes continus, le lecteur peut consulter [Skogestad, 1996].

Vocabulaire de l'ordonnancement	Vocabulaire des systèmes continus
Solution	Loi de commande
Fonction objectif	Critère de performance (précision, rapidité)
Critère de robustesse	Marge de robustesse
$\lambda_k(S, \sigma, I) \leq C_k$	Marge de Module > 0.5

TAB. 1 – Analogie systèmes discrets/systèmes continus

Tout comme en automatique des systèmes continus, la robustesse en ordonnancement dépend des critères que l'utilisateur juge pertinents. Ainsi c'est bien le critère de robustesse qui permet de conclure de la robustesse ou non d'une solution. Ce qui différencie les deux approches est qu'en automatique continue, lorsque les différents critères sont définis, il existe une méthode globale qui permet de conclure sur l'existence ou non d'une loi de commande qui satisfait les critères. En ordonnancement, cette méthode n'existe pas, et chaque problème nécessite une approche spécifique de résolution.

3 CONTEXTE DE L'ÉTUDE ET OBJECTIFS

Nous avons présenté dans la partie précédente l'ordonnancement robuste. Le but de notre travail est justement d'apporter de la robustesse à un atelier particulier de l'unité de production des semi-conducteurs de la manière la moins onéreuse possible. Cet atelier est un parc de machines parallèles partiellement multifonctions [Brucker, 1997]. Il est constitué de m machines. Ces machines sont capables de traiter divers types de produit qui sont au nombre de n. Cependant le traitement de chaque produit demande une mise au point spécifique de la machine et une machine ne peut traiter qu'un ensemble particulier de produits. Lorsqu'une machine est réglée pour traiter un type de produit, on dit qu'elle est qualifiée pour ce type de produits et l'ensemble des qualifications machines/produits constitue la configuration du parc de machines. Enfin de par sa nature même, le parc de machines induit différents coûts : 1) coût de qualification, le fait de permettre à une machine de produire un type de produit implique l'installation d'une résine et le réglage de la machine, 2) coût de maintenance, lors de la production, les résines s'épuisent et doivent être réapprovisionnées.
La configuration de cet atelier est donc très importante et il est indispensable de tenir compte des incertitudes sur la demande pour ainsi assurer une robustesse aux différentes perturbations pouvant survenir dans cet atelier

et dans les ateliers amont. C'est pourquoi chaque mois, la configuration du parc de machines est remise en cause pour faire face à la charge prévisionnelle de la période à venir. Cette charge prévisionnelle est une demande moyenne pour chaque type de produits.
Dans les paragraphes suivants, nous verrons tout d'abord comment nous avons formalisé notre problème, nous définirons ensuite les origines du contexte incertain et enfin nous présenterons le critère de robustesse retenu avec l'industriel.

• *Formalisation du problème statique*
Les données technologiques de ce parc de machines parallèles partiellement multifonctions sont modélisées par une matrice de vitesse notée V de dimension $n \times m$ dont les lignes représentent les types de produit, et les colonnes, les machines. $V(i,j)$ est la quantité de produits de type i que peut traiter la machine j par unité de temps. Si $V(i,j)$ vaut 0, cela signifie que la machine j ne peut pas traiter les produits de type i. C'est une contrainte technologique. De plus les machines sont uniformes : $V(i,j_1) = k_{j_1 j_2} V(i,j_2)\ \forall i$, c.à.d quel que soit le produit considéré, il sera traité $k_{j_1 j_2}$ fois plus vite sur la machine j_1 que sur la machine j_2. Ensuite nous modélisons la configuration du parc de machines par une matrice de qualification notée Q qui a les mêmes dimensions que V. $Q(i,j)$ vaut 1 si la machine j est qualifiée pour le produit de type i et vaut 0 sinon. Cette matrice résulte d'un choix, c'est à dire que même si la machine j peut produire les produits de type i ($V(i,j) > 0$), on peut très bien choisir de ne pas la qualifier pour ce type de produit . Par contre si $V(i,j) = 0$ alors nécessairement $Q(i,j) = 0$. C'est bien cette matrice Q qui est représentative de la configuration réelle du parc de machines et cette configuration implique un coût qui est directement proportionnel au nombre de qualifications (donc de "1") de Q. En effet ce coût est induit par la qualification d'une machine : mise en place des résines, réglage de la machine, maintenance des ressources et de la machine.
La demande prévisionnelle mensuelle de chaque type de produit est représentée par un vecteur colonne noté N_{ref}. Chaque terme représente la quantité de produits de type correspondant à traiter par l'atelier.

• *Exemple académique :*

$$V = \begin{bmatrix} 30,6 & 45,9 & 20,4 \\ 0 & 5,4 & 2,4 \\ 12,6 & 0 & 8,4 \\ 19,2 & 28,8 & 12,8 \end{bmatrix}, Q = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

$$N_{ref} = \begin{bmatrix} 209 \\ 151 \\ 262 \\ 77 \end{bmatrix}$$

• *Origines du contexte incertain*
Dans le secteur d'activité considéré, la demande est soumise à de fortes incertitudes. Tout d'abord le lancement de nouveaux produits est très fréquent et se traduit par un taux de rebut très important en début de vie du produit, ce qui perturbe les prévisions établies pour chaque atelier. Ensuite les performances des machines des ateliers amont dépendent bien souvent directement de leur réglage, la capacité de production de ces ateliers est donc difficile à prévoir, ce qui engendre des perturbations sur l'atelier de photolithographie. Tout ceci implique des incertitudes sur le volume et la répartition des produits à traiter dans cet atelier.

• *Critère de robustesse*
Les machines et leur configuration représentent des coûts d'investissement et de fonctionnement très élevés, l'industriel souhaite que la demande soit satisfaite en équilibrant la charge de travail sur les différentes machines. Le critère de robustesse retenu est donc la maximisation du taux d'occupation des machines. Autrement dit, les machines ne doivent jamais être oisives et doivent finir leur production en même temps malgré les incertitudes sur la demande.
Dans la section suivante, nous verrons comment évaluer la robustesse d'une configuration donnée à l'aide de ce critère.

4 EVALUATION DE LA ROBUSTESSE DE LA CONFIGURATION : PRISE EN COMPTE DE L'INCERTAIN

Dans [Rossi, 2004], les auteurs traitent le même atelier que celui que nous étudions ici.

• *Calcul d'un plan de production optimal, sans prise en compte des incertitudes*
On cherche à calculer un plan de production telle que sa durée totale d'exécution soit minimale. Les auteurs proposent une adaptation du programme linéaire de [Lawler, 1978]. Il permet de trouver une solution optimale en temps polynomial. Le programme linéaire a toujours une solution et un solveur traditionnel peut résoudre des instances de taille industrielle. Le résultat de ce programme linéaire se présente sous la forme d'une matrice de répartition temporelle R ayant les même dimensions que Q. $R(i,j)$ représente le temps passé par la machine j pour traiter le produit de type i. Ainsi une configuration sera robuste si :
$\sum_{i=1}^{i=n} R(i,j_1) = \sum_{i=1}^{i=n} R(i,j_2), \forall j_1 \neq j_2.$
Si Q est la matrice de qualification courante, V la matrice de vitesse et N_{ref} la demande alors le programme linéaire s'écrit :

$$(PL) \begin{cases} min(C_{max}) & \\ \sum_{j=1}^{m} R(i,j) \times V(i,j) \times Q(i,j) = N_{ref}(i) & \forall i \in [1,n] \\ \sum_{i=1}^{n} R(i,j) \times Q(i,j) \leq C_{max} & \forall j \in [1,m] \\ R(i,j) \geq 0 & \forall (i,j) \in [1,n] \times [1,m] \end{cases}$$

• *Prise en compte des incertitudes*
Les travaux [Rossi, 2004] permettent également d'évaluer la robustesse de la configuration vis-à-vis des incertitudes sur la demande. Cette évaluation se traduit par la définition et le calcul d'un rayon de stabilité ([Sotskov, 1998]) qui mesure l'amplitude maximale d'une perturbation sur la demande qui ne remet pas en

cause l'équilibre de la charge dans son traitement par le parc de machines. Ce rayon de stabilité que l'on note $r(Q, N_{ref})$ est calculé pour une configuration et une demande prévisionnelle données. Le calcul de ce rayon de stabilité est présenté en détails dans [Rossi, 2003]. Il est important de noter que le calcul de ce niveau de robustesse passe par l'utilisation de l'algorithme de Nourine et Raynaud ([Nourine, 1999]). Les auteurs ont démontré que le coût de cet algorithme est exponentiel : autrement dit, le temps de calcul peut exploser avec les dimensions de Q, donc avec le nombre de machines et le nombre de types de produit.

⇒ *Application à l'exemple académique*

$$\left.\begin{array}{l} V = \begin{bmatrix} 30,6 & 45,9 & 20,4 \\ 0 & 5,4 & 2,4 \\ 12,6 & 0 & 8,4 \\ 19,2 & 28,8 & 12,8 \end{bmatrix} \\ Q = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} \\ N_{ref} = \begin{bmatrix} 209 \\ 151 \\ 262 \\ 77 \end{bmatrix} \end{array}\right\} \Rightarrow r(Q, N_{ref}) = 55.03$$

Ainsi, pour cet exemple si on a une perturbation dN telle que $\|dN\|_1 \leq 55,03$ alors on peut garantir que la demande $N = N_{ref} + dN$ sera traitée avec un équilibre de la charge pour les machines. Par contre si
$\|dN\|_1 > 55,03$ alors on ne peut rien conclure pour la demande $N = N_{ref} + dN$.
Par rapport à la démarche classique d'optimisation, ces travaux ont permis de caractériser un voisinage de l'instance prévisionnelle du problème dans lequel la performance est garantie.

5 AMÉLIORATION DE LA CONFIGURATION DU PARC DE MACHINES

A l'aide des travaux de [Rossi, 2004], nous sommes capables d'évaluer parfaitement le niveau de robustesse de la configuration du parc de machines par le calcul de son rayon de stabilité vu dans la partie précédente. Dans cette partie nous verrons qu'il est possible de diminuer les coûts d'une configuration tout en préservant son niveau de robustesse.
En effet, si on reprend l'exemple académique, on peut se rendre compte que des qualifications sont inutiles pour avoir un tel niveau de robustesse. Par exemple, si on calcule les rayons de stabilité des configurations définies par :

$$Q = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} \text{ et } Q' = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

on peut se rendre compte qu'ils sont identiques. Les deux configurations ont donc le même niveau de robustesse.
Ainsi, à niveau de robustesse équivalent, on peut avoir une configuration moins onéreuse puisque la configuration représentée par Q' contient trois qualifications de moins que Q.
Donc partant d'une configuration initiale pour laquelle le niveau de robustesse est connu, nous allons chercher à minimiser les coûts (donc trouver le nombre minimum de qualifications) en garantissant le niveau de robustesse initial. Ce niveau de robustesse devient donc une contrainte de notre problème.

5.1 Construction d'une PSE

L'objectif de notre travail est de trouver une méthode permettant de minimiser les coûts liés à la configuration robuste du parc de machines. Pour ce faire, partant d'une configuration initiale, qui est la configuration courante (obtenue de manière empirique par les industriels), nous allons chercher à réduire le nombre de qualifications tout en conservant le niveau de robustesse initial. La procédure que nous choisissons est itérative ; nous cherchons à supprimer une première qualification, puis une deuxième etc. C'est pourquoi nous avons décidé de développer une Procédure par Séparation et Évaluation (PSE).
Nous verrons tout d'abord comment fonctionne notre PSE. Puis nous présenterons les différents outils qui ont été développés pour cette PSE. Enfin nous appliquerons la PSE à un exemple.

5.1.1 Formalisation de notre PSE

La PSE est un outil de l'optimisation combinatoire [Lawler, 1966]. Son objectif est de rechercher une solution (optimale ou non) à un problème, en parcourant de façon intelligente et implicite, et non nécessairement de façon exhaustive, l'ensemble des solutions. Cette démarche se traduit par la construction d'un arbre dont voici la structure (cf. Fig. 1) :
•*Candidat de niveau* k : un candidat de niveau k est un couple $produit/machine$ qui va représenter la k-ième qualification supprimée.
•*Solution* : une solution est un ensemble de qualifications que l'on peut supprimer simultanément sans dégrader le niveau de robustesse.
• *racine* : la racine de l'arbre correspond à la solution vide. Dans notre contexte la solution vide implique qu'aucune qualification n'a été supprimée
•*feuille* : les feuilles sont des éléments de l'ensemble des solutions. Une feuille correspond donc à une solution unique entièrement caractérisée.
• *nœud* : chaque nœud de l'arbre correspond à une solution partielle. Les nœuds qui descendront d'un même nœud ancêtre auront des caractéristiques communes provenant du nœud ancêtre.
• *niveau* : chaque nœud appartient à un niveau dans l'arbre. Le niveau 0 est uniquement composé de la racine. Une feuille est un nœud du niveau le plus élevé dans la branche considérée : aucune décision supplémentaire ne peut être prise (la solution est entièrement construite). Dans notre contexte le niveau k est le niveau dont les nœuds correspondent à des solutions partielles constituées de k qualifications supprimées.

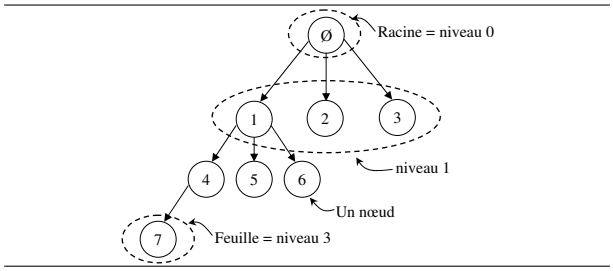

FIG. 1 – un exemple d'arbre

5.1.2 La procédure de séparation

Après avoir vu le schéma général, nous allons maintenant nous intéresser à la procédure de séparation.
Cette procédure permet de construire et de parcourir l'arbre de recherche. Classiquement, il existe deux parcours possibles (même si on peut imaginer des parcours hybrides comme pour notre PSE) :
• *Parcours en profondeur d'abord* : en partant de la racine, une solution partielle de niveau 1 est générée puis évaluée. A partir de cette solution partielle, une solution partielle de niveau 2 est générée et évaluée et ainsi de suite jusqu'à parvenir à une feuille. A partir de cette feuille, on remonte d'un niveau pour réitérer le processus sur une autre branche.
• *Parcours en largeur d'abord* : en partant de la racine, toutes les solutions partielles de niveau 1 sont générées et évaluées. Puis toutes les solutions de niveaux 2 sont générées à partir des solutions de niveau 1 qui n'ont pas été abandonnées. Et on continue jusqu'à ce que tous les nœuds générés soient des feuilles.

5.1.3 La procédure d'évaluation

Après avoir explicité la procédure de séparation, nous allons maintenant décrire la procédure d'évaluation.
A chaque nœud est associée une valeur. La procédure d'évaluation prend toute son importance lorsqu'on cherche une solution optimale dans des délais raisonnables. Le choix de la valeur associée à chaque nœud devient primordial puisque c'est elle qui va permettre d'abandonner des branches non prometteuses. Dans notre cas la procédure d'évaluation est triple puisque 3 valeurs sont associées à chaque nœud : le niveau atteint k, le nombre l de candidats de niveau $k+1$ associé à ce nœud et enfin une borne supérieure BS qui sera explicitée dans la partie suivante.
Une borne inférieure triviale à ces deux dernières valeurs est le plus haut niveau déjà atteint noté n_{max}. Pour la première branche $n_{max} = 0$.
Le nœud est abandonné si $l \leq n_{max} - k$ ou si $BS \leq n_{max} - k$.

5.2 Des outils pour notre PSE

Afin d'optimiser notre PSE, deux outils ont été mis en place :
- Adaptation de l'algorithme de Nourine et Raynaud.
- Calcul d'une borne supérieure.

• *Adaptation de l'algorithme de Nourine et Raynaud*
Chaque nœud de l'arbre que l'on génère pour résoudre notre problème représente une nouvelle qualification supprimée dans la configuration. Cette suppression ne doit pas dégrader le niveau de robustesse de la configuration initiale. Le seul moyen de le vérifier est de calculer le niveau de robustesse de la configuration après suppression de la qualification. Ainsi, pour chaque nœud il est nécessaire de calculer le niveau de robustesse de la configuration courante et donc d'utiliser l'algorithme de Nourine et Raynaud. Or cet algorithme n'est pas polynomial. Lorsqu'on sait que dans la réalité le nombre de machines est supérieur à 10 et que le nombre de types de produit est supérieur à 35, on imagine aisément que le temps peut exploser pour générer tout l'arbre. Il est donc nécessaire de réaliser des pré-calculs pour accélérer la procédure de résolution avec l'algorithme de Nourine et Raynaud. Ces pré-calculs vont consister à réduire la matrice de qualification.
Après avoir étudié le fonctionnement de l'algorithme de Nourine et Raynaud, nous avons pu définir quelques règles qui permettent de réduire la matrice de qualification. En effet les informations nécessaires pour calculer le niveau de robustesse sont contenus dans l'ensemble des rectangles de "0" de Q (cf. [**?**]). On ne perd donc pas ces informations en fusionnant des lignes ou des colonnes identiques ou encore en permutant les lignes ou les colonnes de la matrice Q. Une fois la matrice réduite à l'aide des règles définies ci-dessous, on applique l'algorithme de Nourine et Raynaud sur cette matrice réduite et après traitement du résultat obtenu on retrouve le niveau de robustesse initial. Voici donc ces quelques règles :
- On peut fusionner les lignes identiques de Q.
- On peut fusionner les colonnes identiques de Q.
- On peut transposer Q.
- On peut supprimer itérativement les lignes ou les colonnes pleines de "0" (sous réserve que l'on sauvegarde cette information) ou de "1" jusqu'à obtenir une matrice irréductible.

• *Calcul d'une borne supérieure*
On a vu que pour que la procédure d'évaluation soit performante une borne supérieure doit être définie. Elle doit être la plus efficace possible. Nous allons expliquer ici comment nous avons calculé une borne supérieure pour notre PSE.
Le but de notre PSE est de supprimer le plus de qualifications possible en préservant le niveau de robustesse initial. Notre borne supérieure doit donc être une borne supérieure pour le nombre de qualifications que l'on peut supprimer dans la configuration initiale.
Les informations nécessaires au calcul du niveau de robustesse sont contenus dans l'ensemble des rectangles de "0" de Q. En particulier chaque ligne et chaque colonne constitue un rectangle de "0" de Q. Chaque ligne et chaque colonne de Q contient donc moins d'informations que la matrice entière et ainsi le niveau de robustesse de la configuration générale est plus faible

que le niveau de robustesse de la configuration en ne gardant qu'une machine ou qu'un type de produits. Partant de là, on va considérer chaque sous-configuration qui sera soit constituée d'un seul produit (on supprime toutes les autres lignes dans Q), soit d'une seule machine (on supprime toutes les autres colonnes dans Q). Et sur la sous-matrice correspondante on va supprimer le plus de qualifications possibles, sans dégrader le niveau de robustesse que l'on veut garantir, en triant les "1" de cette sous-matrice selon la vitesse et le volume de la demande. On obtient donc une borne supérieure pour le nombre de qualifications que l'on peut supprimer sur chaque ligne et sur chaque colonne de Q. Si on additionne ces bornes pour les lignes puis pour les colonnes on obtient deux bornes supérieures pour le nombre de qualifications que l'on peut supprimer dans la configuration initiale. Notre borne supérieure finale sera le minimum entre ces deux bornes.
⇒ *Complexité du calcul de la borne* :
Le coût du calcul de cette borne est en $\bigcirc(nm)$. L'algorithme de calcul de notre borne supérieure est donc polynomial.
⇒ *Intérêt supplémentaire de notre borne* :
Nous avons eu envie d'utiliser les bornes sur les lignes et sur les colonnes pour accélérer notre PSE. Après plusieurs expérimentations, nous nous sommes aperçus que le fait de classer les candidats par somme croissante des bornes sur la ligne et la colonne auxquelles ils appartenaient accélérait le processus de résolution. C'est pourquoi, à chaque niveau les candidats sont rangés selon cette règle.

Afin de faciliter la compréhension, nous déroulons notre PSE sur un exemple, dans le paragraphe suivant.

5.3 Application de notre PSE sur l'exemple académique

• *Notations :*
- val valeur qui permet de ranger les candidats à chaque niveau.
- BS borne supérieure pour le nombre de qualifications pouvant être supprimées.
- n_{max} meilleur niveau atteint initialisé à 0.
- (i, j) candidat : i =n° ligne et j =n° colonne.
• *Application à l'exemple académique*
On applique notre PSE à l'exemple académique, on obtient l'arbre de la figure Fig. 2.
Au niveau 0 (cf. Fig. 2) aucune qualification n'est supprimée. La configuration est donc :

$$Q = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

Il y a 9 candidats possibles (tous les "1" de Q) et la borne supérieure est égale à 3 (on ne pourra pas supprimer plus de 3 qualifications simultanément). Pour obtenir le niveau 1, on teste ces 9 candidats.
Le niveau 1 est constitué des candidats qui ne dégradent pas le niveau de robustesse si ils sont supprimés. Les nœuds qui sont barrés à l'aide d'une croix sur la figure Fig. 2 représentent les candidats qui ont été abandonnés, soit parce qu'ils dégradaient le niveau de robustesse, soit parce que la borne supérieure ou le nombre de candidats associés ne permettait pas d'améliorer la meilleure solution. Le niveau 2 est obtenu de la même manière que le niveau 1 à partir du bon candidat situé le plus à gauche c'est à dire le couple (4,2). La configuration devient :

$$Q' = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

On recherche tous les candidats potentiels, on leur associe le niveau de robustesse de la configuration lorsqu'ils sont supprimés ainsi que la borne supérieure de la configuration correspondante et également le nombre de candidats pour le niveau suivant. On se rend compte que tous les candidats doivent être abandonnés car ils dégradent le niveau de robustesse lorsqu'ils sont supprimés. La première solution est donc uniquement composée de $(4, 2)$.
A partir de cette feuille on remonte d'un niveau pour parcourir une autre branche. On développe donc la branche sous $(1, 2)$. La configuration est donc :

$$Q' = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

On obtient le niveau 2 de la même manière que l'on avait obtenu le niveau 1. On évalue les candidats, puis on développe la branche sous le premier bon candidat (c'est à dire $(1, 3)$) pour obtenir le niveau 3. La configuration devient :

$$Q' = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

Le niveau 3 n'est constitué que d'un seul candidat. Après évaluation, il s'avère que ce candidat peut être supprimé sans dégrader le niveau de robustesse. On obtient une deuxième feuille car on ne pourra pas descendre plus bas faute de candidats. De toute façon, la borne supérieure nous garantissait que l'on ne pourrait pas aller plus loin. Nous avons donc une deuxième solution caractérisée par les qualifications $(1, 2); (1, 3); (4, 3)$. Ces 3 qualifications peuvent être supprimées simultanément sans dégrader le niveau de robustesse initial. Et notre PSE peut s'arrêter là car la borne supérieure est atteinte, ce qui signifie que cette solution ne pourra pas être améliorée. Une configuration optimale est donc :

$$Q' = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Nous allons maintenant présenter les expérimentations que nous avons menées pour valider notre approche.

6 EXPÉRIMENTATION

Nous avons testé notre PSE sur une centaine d'exemples. Chaque exemple est constitué d'une matrice de qualification Q, une matrice de vitesse V et une demande de référence N_{ref}. Les matrices Q et V sont générées aléatoirement et N_{ref} est calculée de telle sorte que son traitement par Q soit équilibré (critère de robustesse respecté).
Les exemples sont répartis de la manière suivante :
• Il y a d'une part 54 exemples que nous qualifions

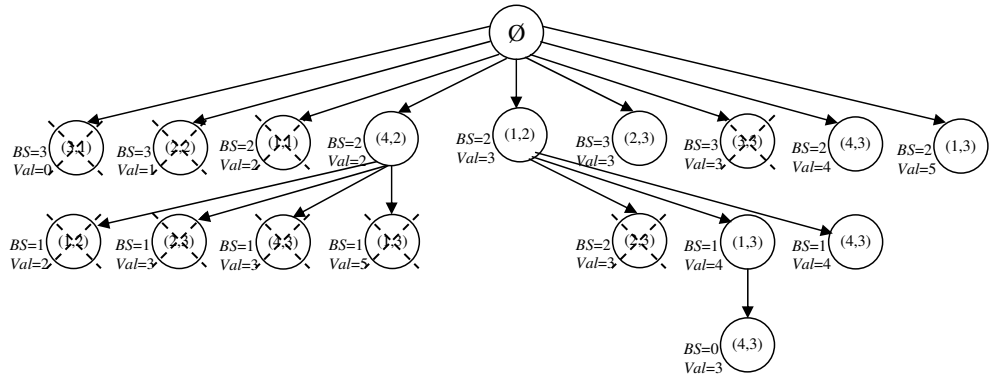

FIG. 2 – Arbre pour l'exemple académique

d'académiques. Ce sont les exemples de type 1. Ils sont caractérisés par une matrice de qualification dont la taille varie entre 4×3 (4 types de produit et 3 machines) et 9×5. Ces exemples de petite taille permettent de tester la justesse et la rapidité de notre PSE.

• Il y a d'autre part 54 exemples de taille industrielle dont la taille de Q varie entre 10×6 et 50×10. Ce sont les exemples de type 2. Ces exemples permettent de déterminer les limites de notre PSE.

Chacun de ces deux panels d'exemples est composé de trois groupes : un groupe dans lequel les matrices de qualification sont *creuses* (plus de 0 que de 1), un deuxième groupe dont les matrices sont *équilibrées* (sensiblement autant de 0 que de 1) et enfin un dernier groupe dont les matrices sont *pleines* (plus de 1 que de 0). Notre PSE a de plus été testée sur un exemple réel constitué de 13 machines et 36 types de produits. Cet exemple constitue à lui seul les exemples de type 3.

Un délai maximal d'exécution arbitrairement égal à une heure a été fixé. Notre PSE a été implémentée en langage C et testée sur un ordinateur dont les caractéristiques sont les suivantes : processeur Mobile Intel Celeron 2,5GHz et RAM de 512Mo.

Les résultats sont présentés dans les tableaux Tab. 2, Tab. 3 et Tab. 4.

Le premier tableau présente les résultats temporels. Ils donnent les temps totaux d'exécution, et les temps pour atteindre la meilleure solution (soit une solution optimale, soit la solution qui contient le plus de qualifications après une heure d'exécution). Le tableau Tab. 3 donne le pourcentage d'exemples résolus optimalement (on garantit qu'une solution optimale a été trouvée car l'ensemble des solutions a été parcouru entièrement). Le dernier tableau (Tab. 4) permet de mesurer la qualité des solutions (lorsque les exemples ne sont pas résolus optimalement) ou la qualité de la borne (lorsque les exemples sont résolus optimalement). Dans ce dernier tableau, seuls les exemples de type 2 et 3 sont considérés car tous les exemples de type 1 ont été résolus optimalement par la PSE.

La première colonne de chaque tableau indique les exemples concernés. Les champs *min*, *moy*, *max* signifient respectivement minimum, moyenne et maximum.

Les résultats temporels (cf. Tab. 2) pour les exemples de type 1 permettent de conclure que 100% des échantillons sont résolus optimalement. Plus de 81% de ceux-ci impliquent un temps d'exécution inférieur à 50ms.

En ce qui concerne les exemples de type 2, notre PSE ne garantit pas toujours que la meilleure solution trouvée est optimale (le délai fixé à 1 heure a été atteint) : seuls 20% des échantillons sont résolus de manière optimale de façon garantie. Cependant la meilleure solution trouvée est de bonne qualité puisqu'elle est supérieure en moyenne à 80% de la borne supérieure (cf. Tab. 4). L'analyse de ce troisième tableau permet de conclure que notre borne n'est pas aussi satisfaisante qu'elle le devrait. En effet, les solutions optimales sont à plus de 20% en moyenne de la borne. Par contre cela veut aussi dire que les solutions approchées, qui sont à moins de 20% de la borne en moyenne, sont de meilleure qualité que la borne ne le suggère.

Lorsqu'on s'intéresse de manière plus précise à la réduction des coûts (cf. Tab. 4) de la configuration initiale (ce qui correspond à l'objectif de notre PSE), on peut se rendre compte que les coûts ont toujours été réduits au moins de $1/3$ et qu'ils sont réduits en moyenne de la moitié.

L'application de notre PSE sur l'exemple industriel (type 3 dans les tableaux) ne permet pas de garantir que la solution trouvée est optimale. Cependant, la meilleure solution trouvée est de très bonne qualité puisqu'elle est éloignée de 15% de la borne supérieure et le coût de la configuration a été quasiment divisée par deux.

A la lumière de ces résultats, on peut dire que si notre PSE ne fournit pas toujours la solution optimale sur des configurations de taille industrielle, elle est tout de même une très bonne heuristique puisqu'elle garantit de très bons résultats (éloignés de moins de 20% en moyenne de la borne) en des temps très raisonnables ($<$ 1heure) quand on sait que le calcul de la configuration se fait mensuellement et hors ligne.

Type d'exemples		Total			Meilleure solution		
		min	moy	max	min	moy	max
Type 1	générale	<1ms	36ms	578ms	<1ms	11ms	109ms
	creuse	<1ms	19ms	172ms	<1ms	10ms	94ms
	équilibrée	<1ms	30ms	390ms	<1ms	9ms	78ms
	pleine	<1ms	60ms	578ms	<1ms	16ms	109ms
Type 2	générale	0,516s	2902s	1 h	0,047s	326s	3245s
	creuse	0,516s	2409s	1 h	0,094s	353s	1993s
	équilibrée	2,36s	3248s	1 h	0,047s	277s	1514s
	pleine	1 h	1 h	1 h	0,968s	347s	3245s
Type 3			1 h			40,7s	

TAB. 2 – Résultats temporels

Type d'exemples		nombre d'exemples résolus optimalement
Type 1	générale	100%
	creuse	100%
	équilibrée	100%
	pleine	100%
Type 2	générale	20%
	creuse	33,33%
	équilibrée	16,67%
	pleine	11,11%
Type 3		0%

TAB. 3 – Pourcentage d'exemples résolus optimalement

Exemples concernés		Qualité.*			Réduction du coût.**		
Type	résolution	min	moy	max	min	moy	max
Type 2	générale				36%	56%	78%
	optimale	69%	78%	92%	36%	47%	59%
	approchée	69%	83%	97%	37%	56%	78%
Type 3	approchée		85%			45%	

*** : nombre de qualifications données par la solution divisé par la borne supérieure**

**** : nombre de qualifications données par la solution divisé par le nombre de 1 de la matrice d'origine**

TAB. 4 – Résultats qualitatifs des solutions

7 CONCLUSION ET PERSPECTIVES

Nous avons étudié un système dynamique de configuration d'un parc de machines parallèles partiellement multifonctions. Après avoir modélisé notre problème nous avons utilisé des propriétés telles que la robustesse pour évaluer la performance d'une configuration du parc de machines. La mesure du niveau de robustesse à l'aide du rayon de stabilité nous a amené à considérer le problème de compromis entre robustesse et coût pour notre atelier. Ce problème omniprésent dans le milieu industriel nous a conduit à utiliser des méthodes issues des mathématiques appliquées (recherche opérationnelle notamment) pour tenter de minimiser les coûts d'une configuration tout en garantissant le niveau de robustesse de départ. L'analyse de nos résultats nous permet de conclure que notre approche répond au problème industriel posé puisque pour l'exemple industriel réel le coût a été divisé quasiment par deux.

L'approche présentée dans cet article est purement hors-ligne (avant mise en production). Si notre approche garantit des performances sur un ensemble de perturbations pouvant survenir pendant la production, ces performances sont uniquement garanties sur un mois. Il serait intéressant de considérer le problème de stabilité d'un mois sur l'autre : c'est à dire maîtriser le coût de la reconfiguration lorsque l'on considère le problème à horizon glissant de la configuration sur plusieurs mois successifs.

BIBLIOGRAPHIE

[Baker, 1974] Baker K., *Introduction to sequencing and scheduling*. Wiley, New York.

[Billaut, 2005] Billaut J.-C., Moukrim A. et Sanlaville E., *Flexibilité et robustesse en ordonnancement*. Hermès, Paris.

[Brucker, 1997] Brucker P., Jurisch B. et Krämer A., Complexity of scheduling problems with multi-purpose machines. *Annals of Operational Research*, 70 :57–73.

[Esquirol, 1999] Esquirol P. et Lopez P., *L'ordonnancement*. Economica, Paris.

[Kouvelis, 1997] Kouvelis P. et Yu G., *Robust Discrete Optimization and its applications*. Kluwer Academic Publisher, Dordrecht.

[Lawler, 1978] Lawler E. et Labetoulle J., On preemptive scheduling of unrelated parallel processors by linear programming. *Journal of the Association for Computing Machinery*, 25 :612–619.

[Lawler, 1966] Lawler E. et Wood D., Branch-and-bounds methods : A survey. *Operations Research*, 14 :699–719.

[Nourine, 1999] Nourine L. et Raynaud O., A fast algorithm for building lattices. *Information Processing Letters*, 71 :199–204.

[Pinedo, 2002] Pinedo M., *Scheduling : Theory, Algorithms and Systems*. Prentice-Hall, Upper Saddle River.

[Rossi, 2003] Rossi A., *Ordonnancement en milieu incertain, mise en oeuvre d'une démarche robuste*. PhD thesis, Institut National Polytechnique de Grenoble.

[Rossi, 2004] Rossi A., Jacomino M. et Espinouse M.-L., Etude de robustesse : configuration d'un parc de machines partiellement multifonctions. *Journal Européen des Systèmes Automatisés*, 34 :373–395.

[Skogestad, 1996] Skogestad S. et Postlethwaite I., *Multivariable Feedback Control : Analysis and Design*. John Wiley & Sons, New York.

[Sotskov, 1998] Sotskov Y., Wagelmans A. et Werner F., On the calculation of the stability radius of an optimal or an approximate schedule. *Annals of Operational Research*, 83 :213–252.