NNT : 2012 EMSE 0650

THÈSE

présentée par

Ali OBEID

pour obtenir le grade de
Docteur de l'École Nationale Supérieure des Mines de Saint-Étienne

Spécialité : Génie Industriel

Ordonnancement et contrôle avancé des procédés en fabrication de semi-conducteurs

soutenue à Gardanne, le 29 Mars 2012

Membres du jury

Président :	Eric PINSON	Professeur, Université Catholique de l'Ouest, Angers
Rapporteurs :	Marc SEVAUX	Professeur, Université de Bretagne-Sud, Lorient
	Vincent T'KINDT	Professeur, Ecole Polytechnique de l'Université de Tours, Tours
Examinateur(s) :	Alix MUNIER	Professeur, Université Paris 6, Paris
	Lars MOENCH	Professeur, Université de Hagen, Hagen, Allemagne
	Boris BETIENNE	Maître de Conférences, Université d'Avignon, Avignon
Directeur de thèse :	Stéphane DAUZERE-PERES	Professeur, Ecole des Mines de St-Etienne, Gardanne
Co-encadrant :	Claude YUGMA	Chargé de recherche, Ecole des Mines de St-Etienne, Gardanne

Spécialités doctorales :	**Responsables :**
SCIENCES ET GENIE DES MATERIAUX	J. DRIVER Directeur de recherche – Centre SMS
MECANIQUE ET INGENIERIE	
GENIE DES PROCEDES	F. GRUY Professeur – Centre SPIN
SCIENCES DE LA TERRE	B. GUY Maître de recherche – Centre SPIN
SCIENCES ET GENIE DE L'ENVIRONNEMENT	J. BOURGOIS Professeur – Fayol
MATHEMATIQUES APPLIQUEES	E. TOUBOUL Ingénieur – Fayol
INFORMATIQUE	O. BOISSIER Professeur – Fayol
IMAGE, VISION, SIGNAL	JC. PINOLI Professeur – Centre CIS
GENIE INDUSTRIEL	P. BURLAT Professeur – Fayol
MICROELECTRONIQUE	Ph. COLLOT Professeur – Centre CMP

Enseignants-chercheurs et chercheurs autorisés à diriger des thèses de doctorat (titulaires d'un doctorat d'État ou d'une HDR)

AVRIL	Stéphane	MA	Mécanique & Ingénierie	CIS
BATTON-HUBERT	Mireille	MA	Sciences & Génie de l'Environnement	Fayol
BENABEN	Patrick	PR 1	Sciences & Génie des Matériaux	CMP
BERNACHE-ASSOLLANT	Didier	PR 0	Génie des Procédés	CIS
BIGOT	Jean-Pierre	MR	Génie des Procédés	SPIN
BILAL	Essaïd	DR	Sciences de la Terre	SPIN
BOISSIER	Olivier	PR 1	Informatique	Fayol
BORBELY	Andras	MR	Sciences et Génie des Matériaux	SMS
BOUCHER	Xavier	MA	Génie Industriel	Fayol
BOUDAREL	Marie-Reine	PR 2	Génie Industriel	DF
BOURGOIS	Jacques	PR 0	Sciences & Génie de l'Environnement	Fayol
BRODHAG	Christian	DR	Sciences & Génie de l'Environnement	Fayol
BURLAT	Patrick	PR 2	Génie industriel	Fayol
COLLOT	Philippe	PR 1	Microélectronique	CMP
COURNIL	Michel	PR 0	Génie des Procédés	SPIN
DAUZERE-PERES	Stéphane	PR 1	Génie industriel	CMP
DARRIEULAT	Michel	IGM	Sciences & Génie des Matériaux	SMS
DECHOMETS	Roland	PR 1	Sciences & Génie de l'Environnement	Fayol
DESRAYAUD	Christophe	MA	Mécanique & Ingénierie	SMS
DELAFOSSE	David	PR 1	Sciences & Génie des Matériaux	SMS
DOLGUI	Alexandre	PR 1	Génie Industriel	Fayol
DRAPIER	Sylvain	PR 2	Mécanique & Ingénierie	SMS
DRIVER	Julian	DR 0	Sciences & Génie des Matériaux	SMS
FEILLET	Dominique	PR 2	Génie Industriel	CMP
FOREST	Bernard	PR 1	Sciences & Génie des Matériaux	CIS
FORMISYN	Pascal	PR 1	Sciences & Génie de l'Environnement	Fayol
FRACZKIEWICZ	Anna	DR	Sciences & Génie des Matériaux	SMS
GARCIA	Daniel	MR	Sciences de la terre	SPIN
GIRARDOT	Jean-Jacques	MR	Informatique	Fayol
GOEURIOT	Dominique	MR	Sciences & Génie des Matériaux	SMS
GRAILLOT	Didier	DR	Sciences & Génie de l'Environnement	Fayol
GROSSEAU	Philippe	MR	Génie des Procédés	SPIN
GRUY	Frédéric	MR	Génie des Procédés	SPIN
GUY	Bernard	MR	Sciences de la Terre	SPIN
GUYONNET	René	DR	Génie des Procédés	SPIN
HERRI	Jean-Michel	PR 2	Génie des Procédés	SPIN
INAL	Karim	PR 2	Microélectronique	CMP
KLÖCKER	Helmut	DR	Sciences & Génie des Matériaux	SMS
LAFOREST	Valérie	CR	Sciences & Génie de l'Environnement	Fayol
LERICHE	Rodolphe	CR CNRS	Mécanique et Ingénierie	SMS
LI	Jean-Michel	EC (CCI MP)	Microélectronique	CMP
MALLIARAS	George Grégory	PR 1	Microélectronique	CMP
MOLIMARD	Jérôme	MA	Mécanique et Ingénierie	SMS
MONTHEILLET	Frank	DR 1 CNRS	Sciences & Génie des Matériaux	SMS
PERIER-CAMBY	Laurent	PR 2	Génie des Procédés	SPIN
PIJOLAT	Christophe	PR 1	Génie des Procédés	SPIN
PIJOLAT	Michèle	PR 1	Génie des Procédés	SPIN
PINOLI	Jean-Charles	PR 0	Image, Vision, Signal	CIS
STOLARZ	Jacques	CR	Sciences & Génie des Matériaux	SMS
SZAFNICKI	Konrad	MR	Sciences & Génie de l'Environnement	Fayol
THOMAS	Gérard	PR 0	Génie des Procédés	SPIN
TRIA	Assia		Microélectronique	CMP
VALDIVIESO	François	MA	Sciences & Génie des Matériaux	SMS
VIRICELLE	Jean-Paul	MR	Génie des procédés	SPIN
WOLSKI	Krzysztof	DR	Sciences & Génie des Matériaux	SMS
XIE	Xiaolan	PR 1	Génie industriel	CIS

Glossaire :

PR 0	Professeur classe exceptionnelle
PR 1	Professeur 1ère classe
PR 2	Professeur 2ème classe
MA(MDC)	Maître assistant
DR	Directeur de recherche
Ing.	Ingénieur
MR(DR2)	Maître de recherche
CR	Chargé de recherche
EC	Enseignant-chercheur
IGM	Ingénieur général des mines

Centres :

SMS	Sciences des Matériaux et des Structures
SPIN	Sciences des Processus Industriels et Naturels
Fayol	Institut Henri Fayol
CMP	Centre de Microélectronique de Provence
CIS	Centre Ingénierie et Santé

Dernière mise à jour le : 23 août 2011

Acknowledgements

I would like to express my gratitude to all those who gave me the possibility to complete this thesis. I want to thank the Department of logistics and manufacturing sciences (SFL) of the Ecole Nationale Supérieure des Mines de Saint-Etienne at Gardanne for giving me permission to commence this thesis in the first instance, to do the necessary research work and to use departmental facilities. I have furthermore to thank Stéphane Dauzère-Pérès and Claude Yugma who helped me by stimulating suggestions and encouragement in all the time of research and encouraged me to go ahead with my thesis.

I would like also to thank all my colleagues who supported me. I want to thank them for all their help, support, interest and valuable hints. I would like also to thank my family for their precious moral support.

Especially, I would like to give my special thanks to my mother whose unlimited love enabled me to complete this work.

Contents

List of Tables

List of Figures

Résumé étendu de la thèse

0.1 Introduction

Cette partie constitue un résumé étendu de ma thèse en français. Je mettrai l'accent sur la présentation de mes travaux, je donnerai quelques explications sur les modèles mathématiques et sur les tests numériques effectués.
Mes travaux de recherche sont liés au contexte de fabrication de semi-conducteurs. Plus précisément, ils se focalisent sur l'intégration des décisions d'ordonnancement et du contrôle avancé des procédés. L'idée majeure est d'intégrer des objectifs similaires (ou presque), exprimés à différents niveaux (niveau processus de fabrication et niveau contrôle avancé) dans le but de mieux produire avec les qualités exigées et à un coût moindre. Dans ce contexte, des problèmes novateurs et pertinents sont définis et étudiés.

Contexte

L'industrie des semi-conducteurs est l'une des industries les plus complexes existant de nos jours. Le nombre considérable d'opérations et le type de procédé par rapport aux industries manufacturières classiques sont les principales sources de cette complexité. Une plaquette de silicium (*wafer*) subit plusieurs centaines d'opérations consécutives avant qu'elle ne soit prête à l'usage. Ceci est dû au fait que les flux dans la fabrication de semi-conducteurs sont de type *reentrant*, i.e. les plaquettes visitent les mêmes machines plusieurs fois. Ces caractéristiques de l'environnement de fabrication et de nombreux autres aspects, font de l'ordonnancement dans de telles industries une question complexe.

Les procédés de fabrication de semi-conducteurs nécessitent un niveau de précision élevé. Des équipements fiables sont nécessaires et des paramétrages précis doivent être fournis. Cela nécessite pour ce type d'industries, d'avoir un système de contrôle fiable. Le rôle du contrôle avancé des procédés (*Ad-*

vanced Process Control-APC) consiste à s'assurer que chaque processus est effectué suivant le cahier des charges prédéfinis et que chaque matériel est fiable pour traiter les types de produits différents.

En fait, les décisions d'ordonnancement d'une part, et le contrôle avancé des procédés (APC) d'autre part, visent à améliorer l'efficacité globale de la fabrication, et plus précisément à maximiser la cadence et à réduire les coûts de fabrication. Il existe différents types de stratégies d'ordonnancement et de techniques utilisées dans différents types d'industries manufacturières. Néanmoins, les décisions d'ordonnancement dans les industries de fabrication de semi-conducteurs sont plus complexes que celles des industries classiques. La difficulté des décisions vient de l'environnement complexe de fabrication des contraintes de fabrication, du nombre de machines et du coût élevé des machines et leurs besoins d'entretien. Dans l'environnemnt de fabrication de semi-conducteurs, il existe de nombreuses techniques d'ordonnancement étudiées et la recherche est toujours active dans ce domaine. Ces recherches tentent de suivre les énormes progrès dans la technologie et de la demande telle que décrite par la Loi de Moore.

Le contrôle avancé des procédés (APC) est devenu un élément indispensable dans le processus de fabrication des semi-conducteurs avec une demande de qualité et une productivité accrue. L'APC utilise diverses techniques afin de maintenir les processus à un niveau de spécification voulu et de surveiller les équipements pour éviter les défauts sur les lots. Pour atteindre ces objectifs, des techniques telles que les statistiques, la fouille de données, les cartes de contrôle et les boucles de régulation sont utilisées. La mise en oeuvre de ces techniques a montré une amélioration notable du rendement, de la productivité des équipements (*OEE*) et de toute l'unité de fabrication de semi-conducteurs (*fab*). En outre, de nombreux sujets de recherche dans le domaine de l'APC sont actuellement à l'étude et les techniques modernes sont analysées et étudiées comme la Métrologie Virtuelle (*VM*).

De nos jours, les *fabs* utilisent des systèmes d'ordonnancement qui n'ont pas vraiment de relations avec le système de contrôle et vice versa. Si nous considérons les techniques d'ordonnancement et nous étudions ses effets sur la fabrication, nous remarquons qu'elles ont un impact sur l'efficacité de fabrication et aussi sur le système de contrôle. Par conséquent, comment faire collaborer les deux systèmes pour améliorer les décisions d'ordonnancement et de contrôle ? Que gagnerait-on de plus ? Et par la suite, comment intégrer l'ordonnancement et les questions de l'APC.

Par la suite, nous caractériserons dans les chapitres à venir des idées d'intégration d'ordonnancement et du contrôle avancé des procédés.

Plan de lecture

Je commencerai par présenter le cadre d'étude de ma thèse, en expliquant les concepts de base pour faciliter la compréhension des travaux. Le plan de lecture sera le suivant :

- En section 0.2, je décris l'environnement de fabrication de semi-conducteurs qui est le contexte de mon étude. Je détaille un peu les décisions d'ordonnancement et le contrôle avancé des procédés dans la fabrication des semi-conducteurs.
- Les motivations pour l'intégration des décisions d'ordonnancement et du contrôle avancé des procédés sont résumées dans la section 0.3.
- En section 0.4, je décris un problème d'ordonnancement avec contraintes de temps, puis en section 3.5 un deuxième problème d'ordonnancement en intégrant les états de santé des équipements. Je discute également des résultats numériques effectués sur des instances générées aléatoirement.
- Des conclusions et des perspectives sont décrites en section 0.8 et section 0.9 respectivement.

0.2 Fabrication de semi-conducteurs : Ordonnancement et Contrôle Avancé des Procédés

La fabrication La fabrication de puces à base de semi-conducteur est composée de trois grandes parties.

Pré-fabrication

Le prétraitement comprend (May and Sze (2003)): la croissance de silicium par le procédé polycristallin, la découpe en tranches et le polissage. Le wafer obtenu peut être utilisé pour commencer le processus de fabrication de puces qui comprend généralement deux étapes : le *Front-End* et le *Back-End*.

Le processus de ***Front-End***

Le processus de Front-End comprend les procédés de Photolithographie, Gravure, Implantation ionique et Planarisation Chimique et Mécanique (CMP).

Le processus de ***Back-End***

Le processus de Back-End comprend deux principaux procédés : les tests et l'assemblage (*packaging*).

L'ordonnancement

Définition

L'ordonnancement traite de l'affectation de ressources limitées à des tâches au fil du temps. C'est un ensemble de décisions ayant pour but d'optimiser un ou plusieurs objectifs tout en satisfaisant un ensemble de contraintes. Dans tous les problèmes d'ordonnancement, les tâches, les contraintes, les ressources et la fonction objectif sont les principaux composants (Carlier and Chrétienne (1988)).

L'environnement de l'ordonnancement

Le processus de fabrication dans la *fab* peut être vu comme un problème d'ordonnancement complexe de type job-shop impliquant plusieurs ateliers de fabrication, différentes variétés de produits, des temps de *setup*, des processus re-entrants, etc. Il existe dans la littérature plusieurs techniques de résolution de ce type de problèmes. On trouve des heuristiques constructives, de la programmation mathématique, des méthodes à base de voisinage, de l'intelligence artificielle, etc. La Figure 1.2, de Moench et al. (2011) résume les principaux problèmes d'ordonnancement rencontrés dans la *fab* et les méthodes de résolution proposées.

Le Contrôle Avancé des Procédés (APC) Le contrôle avancé des procédés (*Advanced Processs Control*-APC) est d'une importance capitale dans les industries de fabrication de semi-conducteurs. L'APC est important pour le contrôle afin d'améliorer continuellement le rendement des machines, la qualité des produits et la baisse des coûts de fabrication. En général, l'APC améliore la fabrication par les apports suivants :

- Développpement rapide des procédés,

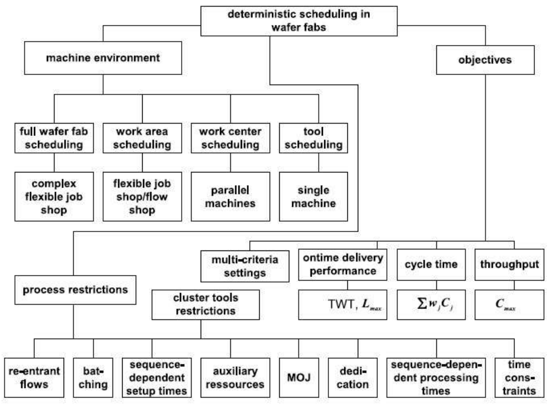


Figure 1: Les problèmes d'ordonnancement déterministes dans la fabs (Moench et al. (2011)).

- Diminution du nombre de wafers moniteurs,
- Diminution de la variation des procédés, augmentation du rendement des machines, meilleure utilisation des boucles de contrôle en cas de dérive,
- Meilleure stabilisation des procédés,
- Augmentation du temps d'utilisation des machines.

L'APC est composée du SPC (*Statistical Process Control* contrôle statistique des processus), du FDC (*Fault Detection and Classification* détection des fautes et classification), du R2R (*Run-to-Run*, régulation) et plus récemment de la VM (*Virtual Metrology*). SPC et FDC se concentrent sur la détection de défauts sur les wafers et les machines ou d'anomalie des procédés. On trouve des techniques de l'APC telles la moyenne mobile pondérée (ou exponentielle), l'analyse des composantes principales (factorielles) qui sont utilisées pour réduire les erreurs et, la *VM* pour prédire des informations de mesure.

0.3 Intégration de l'ordonnancement et du contrôle avancé des procédés

La concurrence s'est accentuée au cours des dernières décennies en raison de la crise financière, des stratégies de croissance, parfois agressives, affichées par certains des acteurs de ce marché, une récession constatée dans certains pays et dans certains secteurs. Pour survivre, les industries sont en permanence à la recherche de solutions idoines qui leur permettraient de rester compétitives. Pour ce faire, une possibilité serait entre autres de partager les informations entre différents systèmes de décision. L'objectif est s'assurer que les décisions prises à un niveau donné n'impactent pas négativement les décisions prises à d'autres niveaux. Cela conduit à des problèmes plus larges et plus difficiles avec des contraintes complexes et des objectifs souvent contradictoires. Dans cette thèse, je m'intéresse à quelques problèmes d'intégration de l'ordonnancement et de l'APC dans l'environnement de fabrication de semi-conducteurs.

0.3.1 Motivations

Nous listons différentes motivations pour l'intégration de l'ordonnancement et de l'APC dans la fabrication de semi-conducteurs.

- Les opérations liées à l'APC (comme par exemple les mesures) peuvent interférer avec les décisions d'ordonnancement en modifiant les ordres de traitement et les listes des opérations à effectuer.

- En utilisant la capabilité des machines fournie en temps réel par des méthodes de l'APC, on peut améliorer les décisions d'ordonnancement, le temps de cycle et le rendement des machines.

- Qualifier (rendre éligible) un équipement a un coût élevé. Ainsi, les algorithmes d'ordonnancement devraient décider comment les opérations seront affectées aux machines qualifiées et équilibrer les charges des machines, le coût d'utilisation de la machine et la qualité du produit obtenu. Dans ce contexte, les machines peuvent être considérées comme qualifiées ou déqualifiées, ou nous pouvons considérer même des niveaux (intermédiaires) de qualification des machines. De plus, une quantité énorme d'informations liées à la machine est recueillie par les systèmes APC, et l'élaboration d'un indice de santé (*Equipment Health Factor*-EHF) permettrait d'optimiser et fiabilier les décisions d'ordonnancement.

- Les paramètres des boucles de contrôle R2R doivent régulièrement être mis à jour. Lorsqu'une machine est utilisée pour la fabrication de différents types de produits, des wafers de test doivent être éxécutés sur les machines pour conserver les boucles de contrôle valides. Par conséquent, un algorithme d'ordonnancement qui prend en compte une telle contrainte sera plus efficace. Cette contrainte peut être présentée par exemple comme une contrainte temporelle liée aux boucles du système de contrôle, ou en nombre de lots qui ont été traités sur la machine. Dans la Section 3.4, une nouvelle contrainte temporelle liée au système APC est présentée et est intégrée dans le problème d'ordonnancement sur des machines parallèles non identiques(présentant différentes qualifications).

Il y a globalement deux façons d'intégrer l'ordonnancement et l'APC :

- Des informations de l'APC pour aider à prendre des décisions d'ordonnancement plus efficaces (section 0.3.2),
- des décisions d'ordonnancement prenant en compte des contraintes de l'APC (section 0.3.3).

0.3.2 Des informations de l'APC pour un ordonnancement plus efficace

Les données disponibles dans le système de contrôle APC peuvent fournir des informations pertinentes pour l'ordonnancement. Ces données peuvent être utilisées comme contraintes ou comme information supplémentaire, pour améliorer les décisions d'ordonnancement. Par exemple, un contrôleur de R2R détecte une erreur (par exemple pas suffisamment de retrait de matière) sur le procédé de planarization (CMP). Un ordonnanceur informé par ces différents taux de polissage pourra envoyer des lots à la machine avec un taux de polissage plus élevé pour compenser le manque de polissage. Ainsi, s'il existe des machines du même type et les conditions de traitement presque similaires, toujours selon les données fournies par les contrôleurs de l'APC, le système d'affectation des lots peut utiliser cette information, pour compenser les dérives.

De plus, le système APC peut fournir un indice de capabilité de la machine ou du procédé(C_{pk}). Il spécifie que la machine est capable de traiter un produit tout en restant conforme aux spécifications. Ces informations fournies par le système APC aident le système d'ordonnancement à sélectionner les machines avec les C_{pk} adéquats pour les couches critiques. Cela permet d'améliorer le rendement. En complément, le système de contrôle APC fournit des informations sur l'état de santé des équipements, qui peuvent être utilisées pour définir un facteur (indice) de santé pour chaque équipement variant au fil du temps. L'ordonnancement va tenir compte des facteurs de santé pour affecter des lots. Et l'acheminement des lots ou des wafers sur les machines deviendrait encore les plus fiable.

Cependant, les machines qui sont affectées à l'exécution de tâches données devraient être qualifiées avant de commencer le procédé. Des lots sont envoyés en éclaireur sur les machines pour vérifier leur conformité avant l'ordonnancement. Cette qualification est faite généralement par l'envoi d'un type spécial de wafers (*Send Ahead Wafers*- SAW) pour tester l'équipement. Cette information peut être utilisée dans l'ordonnancement pour développer des solutions avec une vue plus claire sur l'équipement le plus fiable pour exécuter le procédé, en intégrant le rendement attendu ou la probabilité de perte de puces.
Généralement, lorsqu'un lot a fini son procédé sur une machine, il fait l'objet d'opérations de contrôle en métrologie (par exemple l'épaisseur d'une couche de dioxyde de silicium). Certaines opérations sont donc pour la production et d'autres pour des tests de contrôle. Le système d'ordonnancement devrait alors décider non pas seulement comment est traité le lot, mais également comment il est mesuré (Dauzère-Pérès et al. (2010)). Comment sélectionner les lots pour les mesurer et comment ordonnancer les lots sélectionnés sont des questions intéressantes pour améliorer l'efficacité de fabrication.

L'APC permet de ne pas démarrer le traitement d'un lot sur une machine lorsqu'il n'y a pas d'information dans le système de contrôle, obtenue généralement par les boucles *R2R*. Il est donc important de garder actives (aptes) les boucles *R2R* à valider l'état des machines. Pour que les boucles *R2R* reste valides, il est nécessaire d'envoyer à intervalles de temps régulier des lots sur les machines pour garantir la conformité du produit. Un système d'ordonnancement ayant cette information de l'APC, utilisera cette information pour garder les boucles de contrôle *R2R* mises à jour. Par exemple, sachant que les paramètres d'une boucle *R2R* vont être obsolètes, le système d'ordonnancement programmera les lots de manière à permettre cette mise à jour des paramètres de la boucle *R2R*. Ainsi, les paramètres de la boucle *R2R* sont mis à jour avant qu'ils ne soient hors des spécifications de contrôle (ce qui exige habituellement une recalibration de la machine entraînant des coûts non négligeables). De plus, cela permet des diminutions de temps de cycle en gardant les machines éligibles pour le traitement des lots.

0.3.3 Ordonnancement avec des contraintes de l'APC

L'ordonnancement des lots de production peut améliorer les performances des systèmes de contrôle de l'APC en particulier dans un environnement où il y a plusieurs machines, différents produits et procédés. À titre d'exemple, il peut y avoir des milliers de boucles de contrôle à mettre à jour. Plus

précisément, les recettes ajustant les boucles de régulation R2R exigent des opérations de métrologie régulières à effectuer pour chaque type de produit et de procédé. De plus, lorsque les conditions de traitement ne sont plus aux normes, le risque de perdre un lot/wafer augmente. Pour mieux comprendre, considérons le scénario suivant : une machine traite trois types de produits. Chaque type de produit a un temps limite (seuil) dans lequel il doit être traité, sinon on considère que la machine n'est plus apte à traiter ce type de produits. Ces temps limite sont renouvelés chaque fois que les produits sont exécutés. Lorsque le seuil temporel d'un des produits est dépassé, la machine ne reçoit plus que deux types de produits, le troisième ne peut être traité car il y a non respect du seuil temporel. La régulation de la boucle pour ce troisième produit n'est plus valide. La boucle de contrôle de ce dernier produit doit donc être requalifiée (homologuée). Donc, pour maintenir la qualification, il est alors nécessaire de réaliser le traitement du troisième type de produits avant la fin de l'intervalle de temps. L'ordonnancement qui prendrait en compte cette contrainte améliorerait l'efficacité globale de production (*Overall Effectiveness Equipment-OEE*) et diminuerait le nombre de wafers tests.
Dans certaines zones, il y a un temps maximum autorisé entre des étapes du processus. Par exemple, un matériau qui a été traité à une certaine étape doit passer à l'étape suivante dans un certain temps, sans quoi la procédure devra être répétée. Par conséquent, un système d'ordonnancement qui tient compte de ces contraintes diminuerait les étapes redondantes et par conséquent améliorerait le temps de cycle global et le rendement global de la fab.
Dans ce qui suit, nous étudions deux exemples de problèmes d'ordonnancement avec des contraintes de l'APC.

0.4 Ordonnancement de famille de jobs sur machines parallèles non-identiques et contraintes de temps(PTC)

Court état de l'art L'impact de l'APC sur les décisions de l'APC est analysé par Li and Qiao (2008). Les auteurs étudient également l'ordonnancement de familles de jobs sur des machines parallèles. Ils considèrent que les machines sont identiques, que la qualification des machines peut être ordonnancée et que la limite entre deux jobs d'une même famille est donnée en nombre de jobs. Dans cette thèse, nous considérons des machines parallèles non identiques et nous supposons que les qualifications ne peuvent pas être or-

donnancées et sont effectuées après l'horizon d'ordonnancement. Le problème devient plus complexe, étant donné que l'affectation des jobs aux machines est essentielle pour éviter les déqualifications. Enfin, nous considérons une limite (un seuil) exprimée en temps plutôt qu'en nombre de jobs. Les deux types de seuil sont pertinents et sont liés. Cai et al. (2011) étudient l'interaction entre l'ordonnancement et l'APC sur une machine, avec des temps de setup entre deux familles de jobs et une qualification à effectuer lorsque la contrainte R2R n'est pas respectée. Ils montrent que le problème sur une machine avec plusieurs familles de jobs est NP-difficile. Un autre exemple de l'intégration des contraintes de l'APC dans les décisions d'ordonnancement se trouve dans Detienne et al. (2012), où les opérations de mesures sont idéalement ordonnancées afin de minimiser le risque de perte des produits.
Dans cette thèse, nous abordons deux nouveaux problèmes d'ordonnancement : un problème avec des contraintes de temps (*Problem with Time Constraints-PTC*) et un problème d'ordonnancement avec intégration de la santé des équipements (*Problem with Equipment Health Factor-$PEHF$*). Dans PTC, il y a un temps de contrainte sur les jobs de la même famille, c'est-à-dire que l'intervalle de temps entre deux jobs consécutifs d'une même famille doit être inférieur ou égal à un seuil fixé. L'objectif est d'ordonnancer les familles de jobs sur les machines parallèles avec des informations de l'APC tout en réduisant le nombre de déqualifications des machines. Comme mentionné ci-dessus, la contrainte spécifique de PTC est inspirée des besoins des systèmes de l'APC, et en particulier les boucles $R2R$ pour un type donné de produit sur une machine, qui exigent de recueillir régulièrement des données. $PEHF$ est une extension de PTC dans lequel les états de santé des machines sont pris en compte.

Ordonnancement avec contraintes de temps

Définition

Rappelons que, dans PTC, les lots (jobs) de différentes (familles) sont prévus sur des machines parallèles et toutes les machines ne sont pas capables de traiter toutes les familles de jobs (machines non identiques). Nous considérons une contrainte de temps sur les jobs d'une même famille, c'est-à-dire que l'intervalle de temps entre deux jobs consécutifs d'une même famille doit être inférieur à un seuil de temps donné. Comme il a été déjà mentionné, cette contrainte est inspirée des besoins du système de l'APC et en particulier des boucles R2R, qui nécessitent de recueillir régulièrement des données pour les types de produits sur les machines. Dans notre problème, nous supposons que cette qualification ne peut être effectuée dans l'horizon d'ordonnancement. Le problème devient plus complexe, étant donné que l'affectation des jobs aux machines est essentielle pour éviter les pertes de qualification.

Notre objectif est d'ordonnancer, sur un horizon discrétisé en T périodes, un ensemble de N jobs de différentes familles sur un ensemble de M machines parallèles. Le nombre de familles de jobs est noté F et $f(i)$ est la famille du job i. Nous supposons que les temps de traitement p_f de tous les jobs de la famille f sont égaux. Les machines ne sont pas qualifiées pour traiter toutes les familles de jobs. La qualification d'une machine peut être perdue à un certain instant en raison d'un changement dans le niveau de confiance de la machine. Un temps de setup $s_{f'}$ sur une machine est nécessaire pour changer d'un job d'une famille f à un autre job de famille f', où $f \neq f'$. Enfin, les contraintes de contrôle *R2R* sont considérées à travers un paramètre γ_f, qui correspond à l'intervalle de temps maximal (appelé seuil de temps dans la suite) entre le traitement des deux jobs de la famille f sur une machine qualifiée. Habituellement, si cette contrainte n'est pas satisfaite, une qualification doit être effectuée pour qualifier de nouveau la machine pour la famille f. Par la suite, nous considérons que la machine n'est plus disponible pour tout job de la famille f si la qualification ne peut être maintenue. L'objectif est d'optimiser la somme des dates de fin des jobs, tout en minimisant le nombre de déqualifications des machines. Nous avons donc un problème d'ordonnancement bi-critères. Selon la notation $\alpha|\beta|\gamma$ introduite par Graham et al. (1979) pour classer les problèmes d'ordonnancemnt, ce problème est noté $P_m|ST_{si,b}|\sum c_j$.

Notations

Voici les notations et les paramètres utilisés pour la modélisation du problème.

Paramètres: Les paramètres sont :

T: Nombre de périodes dans l'horizon,
N: Nombre de jobs,
M: Nombre de machines,
F: Nombre de familles de jobs,
$M(f)$: Ensemble de machines qualifiées pour traiter les jobs de la famille f ($M(f) \subset M$),
p_f: Durée d'exécution des jobs de la famille f,

s_f: Temps de setup des jobs de la famille f,
γ_f: Seuil de temps pour les jobs de la famille f,
$f(i)$: famille du job i ($f(i) \in F$).

Variables de décision Les variables de décisions sont :

$x^m_{i,t} = 1$ si le job i commence à la période t sur la machine m, et $= 0$ sinon,
C_f: Somme des dates de fin de tous les jobs de la famille f,
$y^m_{f,t} = 1$ si le seuil de temps n'est pas satisfait pour la famille f sur la machine m à la période t, i.e. une qualification est nécessaire, et $= 0$ sinon.
$Y^m_f = 1$ si le seuil de temps n'est pas satisfait pour la famille f sur la machine m à la fin de l'horizon, et $= 0$ sinon.

Il est important de rappeler que si le seuil de temps γ_f n'est pas satisfait pour le job de la famille f sur la machine m, la qualification sur la machine m ne peut être effectuée pendant l'horizon. Dans ce cas, nous supposons qu'aucun job de famille f ne peut être exécuté sur m. On utilisera les mêmes notations pour la modélisation $PEHF$.

Modélisation basée sur la famille des jobs ($IP3$)

$$\sum_{m \in M(f)} \sum_{t=1}^{T-p_f+1} x^m_{f,t} = n_f \quad \forall f \in F \tag{1}$$

$$\sum_{m \in M(f)} \sum_{t=1}^{T-p_f+1} (t + p_f - 1) x^m_{f,t} \leq C_f \quad \forall f \in F \tag{2}$$

$$\sum_{\tau=t-p_f+1}^{t} x^m_{f,\tau} \leq 1 \quad \forall t = 1 \dots T, \forall f \in F, \forall m \in M(f) \tag{3}$$

$$\sum_{\tau=t-p_f-s_{f'}+1}^{t} x^m_{f,\tau} + n_f . x^m_{f',t} \leq n_f \quad \forall t = 1 \dots T, \forall (f, f') \in F \times F \text{ s.t. } f \neq f', \forall m \in M(f) \cap M(f') \tag{4}$$

$$\sum_{\tau=t-\gamma_f+1}^{t} x^m_{f,\tau} + y^m_{f,t} \geq 1 \quad \forall f \in F, \forall m \in M(f), \forall t = \gamma_f \dots T \tag{5}$$

$$y^m_{f,t-1} - 1 + \frac{1}{T-(t-1)} \sum_{\tau=t}^{T} \sum_{f' \in F} \sum_{m' \in M(f')} x^{m'}_{f',\tau} \leq Y^m_f \quad \forall t = 2 \dots T, \forall f \in F, \forall m \in M(f) \tag{6}$$

$$x^m_{f,t} \in \{0,1\} \quad \forall t = 1 \dots T, \forall f \in F, \forall m \in M(f) \tag{7}$$

$$y^m_{f,t} \in \{0,1\} \quad \forall t = 1 \dots T, \forall f \in F, m \in M(f) \tag{8}$$

$$Y^m_f \in \{0,1\} \quad \forall f \in F, m \in M(f) \tag{9}$$

La fonction objectif

Deux types de critères sont considérés pour PTC. Le premier type correspond à la minimisation de la somme des dates de fin $\sum_{f\in F} C_f$. Le second type est associé au nombre de déqualifications $\sum_{f\in F}\sum_{m\in M(f)} Y_f^m$. Notre fonction objectif est une somme pondérée de ces deux types de critères: $\alpha\sum C_f$ et $\beta\sum Y_f^m$, où α et β sont des poids associés à chacun des critères. Cependant, il semble plus réaliste d'envisager un ordre lexicographique, où le nombre de déqualifications est priorisé sur le critère d'ordonnancement pur, c'est-à-dire β est choisi suffisamment grand par rapport à α ($\alpha = 1$, $\beta = |N| * T$), de sorte que l'amélioration du critère d'ordonnancement n'est pas préférable à une déqualification supplémentaire.

0.5 Ordonnancement avec contraintes de temps et état de santé des équipements ($PEHF$)

Définition

Nous rappelons que le problème PTC est un problème d'ordonnancement des lots (jobs) de différentes familles sur des machines parallèles, où toutes les machines ne sont pas capables de traiter toutes les familles de jobs (machines non identiques). $PEHF$ peut être considéré comme une extension de PTC où nous ajoutons à la contrainte de temps des jobs d'une même famille la notion de risque d'affecter des familles de jobs sur les machines. Le problème devient plus complexe, étant donné que l'affectation des jobs aux machines est essentielle à la fois pour éviter les déqualifications, et pour minimiser/maximiser la probabilité de rendement associée à l'affectation.
Par conséquent, $PEHF$ est un problème d'ordonnancement, avec discrétisation en T périodes, un ensemble de N jobs des différentes familles sur un ensemble de M machines parallèles. Nous rappelons que le temps de traitement p_f de tous les jobs de la famille f est le même et qu'un temps de setup s_f sur une machine est nécessaire pour passer d'un job d'une famille f à un autre job d'une autre famille f, où $f \neq f'$. La contrainte de temps est considérée avec le seuil de temps γ_f. La valeur de v_f^m est considérée comme le rendement attendu résultant de l'affectation d'un job de la famille f à une machine m. L'objectif est d'optimiser un problème d'ordonnancement multicritères (somme des dates de fin, somme des déqualifications des machines et somme des probabilités de perte de lots (somme totale de rendement attendu)).

Maximisation du rendement

La fonction objectif comprend trois critères qui sont la somme des dates de fin, le nombre de déqualifications des machines et la somme du rendement attendu. Nous proposons une somme pondérée de ces trois critères dans notre fonction objectif. L'équation (3.47) représente la fonction objectif à maximiser avec les mêmes contraintes que le modèle $IP3$. Le modèle $IP5$ correspond à $IP3$ avec la fonction objectif suivante :

$$\max\,(\gamma^{'}\sum_{f\in F}\sum_{m\in M}\sum_{t=1}^{T} w_f x^m_{f,t} v^m_f - (\alpha^{'}\sum_{f\in F} C_f + \beta^{'}\sum_{f\in F}\sum_{m\in M} Y^m_f)) \qquad (10)$$

0.6 Expérimentations avec les modèles de programmation mathématique

Génération des instances

Pour tester les modèles de programmation mathématique PTC et $PEHF$ et les heuristiques de la section 0.7, des instances ont été générées aléatoirement. Pour les deux problèmes, les seuils de temps des familles de jobs ont été déterminés suffisamment grands en tenant compte des durées d'exécution. Cela a été fait pour donner un biais minimal pour trouver une solution, étant donné que les seuils courts peuvent entraîner une perte très rapide des qualifications des machines. Et, par la suite, il ne sera pas possible de traiter tous les jobs disponibles, étant donné que les jobs ne peuvent pas être envoyés aux machines déqualifiées. Nous considérons que $Max(p_f) \leq Min(\gamma_f)$. Le schéma de qualification initiale de famille/machine a été défini afin que chaque famille ait au moins une machine sur laquelle elle peut être traitée, et chaque machine est qualifiée pour exécuter au moins une famille. Ceci a pour but d'éviter les solutions irrealisables pour une instance donnée. Les temps de *setup* ont été choisis pas trop grands afin que la probabilité de perte d'une qualification de machine soit acceptable. Nous considérons que $Max(s_f) \leq Min(p_f)$. L'horizon temporel a été pris comme la somme des durée opératoires, plus les temps de *setup* multipliés par le nombre de jobs par famille. C'est un cas extrême où tous les travaux sont prévus sur une seule machine et, chaque fois qu'un job est ordonnancé, un temps de setup est requis ($T = \sum f \in Fn_f * (p_f + s_f)$). Les types d'instances ont été choisis en tenant compte des effets possibles du nombre de jobs, des familles, des machines et aussi de la plus grande durée opératoire.

De plus, la valeur de v_f^m, c'est-à-dire le rendement attendu résultant de l'affectation des jobs de la famille f à une machine m, est considéré comme dominant par machine. Cela signifie que, chaque fois qu'une machine m^* est *en meilleure santé* que m', alors le rendement obtenu lors de l'affectation des jobs de la famille f à m^* est meilleur que l'attribution des mêmes jobs de la famille f à toute autre machine m', c'est-à-dire $v_f^{m^*} \geq v_f^{m'}, \forall f \in F$. Cette hypothèse est forte, elle hiérarchise le facteur de santé de la machine sur tout autre facteur comme par exemple la criticité du lot/famille comme indiqué à la section 0.3. Nous pensons que cette hypothèse, en plus de l'existence de γ_f, ajoute plus d'importance à l'EHF en rendant la conséquence d'une décision d'ordonnancement (affectation d'une machine à une famille de jobs) plus dépendant de la machine. D'autre part, la notion d'indice de criticité famille lot/wafer est toujours considéré dans les valeurs de rendement.

Analyse des fonctions objectif PTC et $PEHF$

Problème avec contrainte de temps (PTC)

Dans ce problème, les deux critères de la fonction objectif de PTC sont optimisés en utilisant les valeurs suivantes: $\alpha = 1$ et $\beta = k * \beta_o$, où $\beta_o = |N| * T$ et $k = 1/\beta_o$, 1/6, 1 /4, 1/2 et 1. Ces valeurs sont choisies de manière à ce que le critère d'ordonnancement, i.e. la minimisation de la somme des dates de fin, soit le deuxième critère important, et la minimisation du nombre de déqualifications soit le premier critère sauf pour $k = 1/\beta_o$; de sorte que c'est équivalent à un ordre lexicographique. Le poids du critère de qualification est varié en fonction de β_o, ceci pour étudier l'effet de la variation des valeurs de la fonction objectif. Les résultats montrent que le problème devient beaucoup plus difficile avec l'augmentation du poids du critère de qualification, de même qu'avec $k = 1/6$. Cela est dû au fait que l'objectif est non seulement de trouver un ordonnancement possible pour maximiser la satisfaction des contraintes seuil, mais aussi de mettre en balance la minimisation de la somme des dates de fin. En revanche, pour les types d'instance avec un temps de résolution inférieur à la limite de temps, les valeurs de $\sum_f \sum_m Y_f^m$ sont plus petites car la valeur du poids de ce critère a augmenté. Toutefois, les valeurs de $\sum_f C_f$ augmentent. C'est un comportement normal et la nature antagoniste des deux critères est mise en évidence.

Le pourcentage d'augmentation de la somme des pertes des qualifications des machines a tendance à être plus petit pour les petites valeurs de β. Notez que ce n'est pas garanti pour les valeurs plus élevées de β, car l'augmentation en pourcentage de la somme des dates de fin est directement proportionnelle à cette variation lorsqu'il n'y a aucune domination complète d'un critère sur l'autre, c'est-à-dire des vecteurs de préférence (1,$\beta_o/2$), (1,$\beta_o/4$) et (1,$\beta_o/6$). La somme des pertes des qualifications des machines est maximale pour le vecteur de préférence (1, 1) où la somme des dates de fin est la priorité et est minimale pour le vecteur de préférence (1, β_o)), où la déqualification des machines est priorisée.

Problème avec état de santé des équipements ($PEHF$)

Dans ce problème, les trois critères de la fonction objectif de $PEHF$ sont optimisés, en utilisant les valeurs suivantes: $\alpha^{'} = 1$, $\beta^{'} = \beta_o = |N| * T$ et $\gamma^{'} = k{*}\gamma_o$, où $\gamma_o = |M|{*}|F|{*}|N|{*}T$ et $k = 1/\gamma_o$, 1/8, 1 /6, 1 /4, 1/2 et 1. Ces valeurs sont choisies de manière à ce que le critère d'ordonnancement (somme des dates de fin) est moins priorisé par rapport à la minimisation du nombre de déqualifications et la somme totale du rendement attendu. Le poids du critère de rendement est augmenté en fonction de γ_o, pour étudier l'effet de $\gamma^{'}$ sur la fonction objectif. Les essais sont réalisés tout en modifiant le coefficient de multiplication du critère de rendement. Ce critère est prépondérant car $PEHF$ est considéré comme une extension de PTC, et donc minimiser les déqualifications devrait toujours être pris en compte. Les résultats montrent que, pour les grandes valeurs de k, le rendement est mieux optimisé pour tous les types d'instance dans le temps limite de calcul.

Les valeurs de α' et β' sont respectivement fixés à 1 et β_o pour étudier l'effet de la variable γ sur les différents critères de la fonction objectif. La valeur de β est fixée à β_o, parce que $PEHF$ est une extension de PTC et que les qualifications de la machine sont importantes. Les valeurs de β_o sont utilisées pour tester la nature antagoniste des qualifications des machines et du rendement attendu. Le pourcentage de gain sur le rendement attendu pour les instances sélectionnées apparaît fondé sur les valeurs obtenues pour le vecteur de préférence (1,β_o,1), où les qualifications des machines sont priorisées. Le pourcentage obtenu pour le vecteur de préférence (1,β_o,γ_o), c'est-à-dire priorisant la qualification des machines et le rendement attendu, augmente par rapport aux autres vecteurs de préférence où la priorité du rendement prévu décroît progressivement ($\gamma = \gamma_o/2$, $\gamma_o/4$, $\gamma_o/6$ et $\gamma_o/8$). Une fois que le poids du rendement prévu est diminué, et que le poids des qualifications des machines est augmenté, le pourcentage de gain sur la diminution du rendement attendu et le nombre des qualifications des machines diminue aussi.

De plus, l'effet de variation du poids associé aux rendements attendus par rapport à la somme des dates de fin est également étudié. Les résultats montrent que la tendance est une augmentation de la somme des dates de fin lorsqu'on priorise le rendement attendu. Cela illustre une fois encore la nature antagoniste des deux critères. Cependant, ce n'est pas toujours vrai puisque la somme des dates de fin n'est pas toujours antagoniste avec le rendement attendu. Afin de minimiser la somme des dates de fin, il y a une tendance à ordonnancer les jobs d'une famille donnée sur la même "meilleure" machine afin de réduire les setups et donc la somme des dates de fin.

Frontières de Pareto pour *PTC*

Le problème *PTC* est un problème à deux objectifs (contraditoires). Pour les problèmes à plusieurs objectifs, on s'intéresse souvent à déterminer des solutions Pareto-Optimales. *PTC* comprend deux critères : un critère d'ordonnancement classique qui est la somme des dates des jobs et un critère particulier qui est la somme des déqualifications de la machine. Les résultats des tests montrent qu'il existe un fort compromis entre le maintien des qualifications des machines et l'exécution dès que possible des jobs. De plus, les résultats indiquent que la somme des dates de fin peut augmenter considérablement lorsque le nombre maximal de pertes de qualification de la machine est réduit et vice versa.

Analyse de sensibilité du seuil de temps pour *PTC*

L'impact des seuils de temps sur la somme des pertes de qualification des machines est également étudié. Dix types d'instances différentes sont choisies et analysées. Le seuil de temps de la première famille varie de 1 à l'horizon T, tandis que les seuils des autres familles sont définis à T pour mettre l'accent sur le seuil d'une famille à la fois c'est-à-dire, $\forall f \in F, f \neq f_1 \gamma_f =$ T. Les tests sont effectués tout en priorisant le critère de qualification ($\alpha = 1$ et $\beta = |N| * T$) pour étudier l'effet du seuil de temps sur ce critère. Les résultats montrent que le nombre de déqualifications des machines diminue à mesure que le seuil de temps augmente. Les machines ont plus de temps pour traiter les jobs de la famille et passer à d'autres familles.

0.7 Heuristiques et Metaheuristiques

Heuristiques constructives

Des heuristiques sont développées pour résoudre *PTC* et *PEHF* : (Scheduling-Centric Heuristic *SCH*) et (Qualification-Centric Heuristic *QCH*). De plus, (Yield-Centric Heuristic *YCH*) est également développé pour traiter le critère de somme totale de rendement attendu de $PEHF$. La première heuristique (SCH), comme son nom l'indique, met l'accent sur l'ordonnancement, la deuxième (QCH vise à réduire au minimum la perte de qualifications des machines et la troisième vise à maximiser le rendement total (YCH).

Scheduling-Centric Heuristic (SCH)

Le principal objectif de SCH consiste à minimiser les temps de *setup*. Rappelons qu'un temps de setup est nécessaire lorsque deux jobs de différentes familles sont ordonnancés consécutivement sur une machine. L'heuristique ordonnance les jobs d'une même famille successivement sur une machine, jusqu'à ce que tous les jobs de la famille soient ordonnancés ou qu'il ne soit plus possible d'ordonnancer les jobs de la famille sans perdre la qualification de la machine. S'il n'est plus possible d'ordonnancer les jobs d'une famille avant son seuil de temps restant, la machine est déqualifiée pour la famille.

Qualification-Centric Heuristic (QCH)

L'objectif principal QCH est de minimiser le nombre de violations de la contrainte de temps sur les machines sur lesquelles chaque famille est qualifiée. Pour ce faire, dans la première phase de QCH, les jobs avec le seuil de temps restant le plus petit sont ordonnancés sur une machine en priorisant toujours la qualification. Cependant, cela entraîne un grand nombre de setups. C'est pourquoi deux autres phases d'améliorations locales sont appliquées pour réduire le nombre de setups. La phase 2 tente de faire avancer le dernier job de chaque machine en le combinant avec le premier job de la même famille ordonnancée sur la machine, c'est-à-dire des changements intra-machines sont évalués. La phase 3 permet des changements inter-machines.

Yield-Centric Heuristic (YCH)

L'heuristique (YCH) vise à maximiser le rendement total

$$\sum_{f\in F}\sum_{m\in M}\sum_{t=1}^{T} x_{f,t}^{m} v_{f}^{m}$$

Elle ordonnance d'abord les jobs sur la machine ayant un rendement maximum. Ensuite, l'heuristique tente de garantir la non violation des seuils de temps.

Les heuristiques SCH et QCH

En comparant les heuristiques entre elles, on remarque que les résultats de SCH sont le plus souvent meilleurs que ceux de QCH en termes de somme des dates de fin (38/47 instances résolues). D'autre part, comme prévu, QCH donne les meilleurs résultats en termes de pertes de qualification des machines (49/49 instances résolues). Il est important de noter que les heuristiques ne peuvent pas toujours résoudre les instances car, selon l'heuristique, une machine peut perdre sa qualification pour une certaine famille et aucun traitement sur une autre machine n'est possible (aucune autre machine qualifiée pour cette famille de jobs). Dans ce cas, les jobs de la famille considérée ne pouvent être ordonnancés sur une autre machine et la solution est irrealisable. C'est le cas avec les instances 6 pour QCH, les instances 6, 23 et 29 pour SCH.

Les heuristiques SCH, QCH et YCH

Les simulations ont montré que, pour certaines instances, les solutions peuvent être réalisables pour certaines heuristiques et irréalisables pour d'autres. YCH a la meilleure solution pour la somme du rendement attendu dans 41/47 instances résolues comparé à SCH et QCH. QCH a les meilleures solutions concernant la somme des déqualifications des machines pour 45 /49 instances résolues par rapport à SCH et YCH. Quant à la somme des dates de fin, SCH a de meilleures performances pour 29/47 instances. Cependant, YCH montre une performance différente pour la somme des dates de fin (les meilleures solutions sont obtenues pour 26/49 instances). Cela s'explique par la tendance de YCH à ordonnancer les jobs de la même famille sur la machine avec le meilleur indice de santé, minimisant ainsi le nombre de setups, et par conséquent la somme des dates de fin.

Heuristique Récursive (RH)

L'idée générale de cet algorithme est d'ordonnancer des tâches en acceptant chaque fois une perte de qualification ou plus. Plus précisément, nous envisageons une solution obtenue par une des heuristiques introduites précédemment, et nous réappliquons cette heuristique après un changement du schéma de qualification initiale. Les perturbations dans le schéma de qualification sont choisies parmi l'ensemble des machines qui ont perdu leurs qualifications dans la solution obtenue par l'heuristique. En d'autres termes, avec un ordonnancement des jobs utilisant une heuristique donnée, nous étudions la solution afin de vérifier si les machines ont encore la capacité de traiter les jobs, c'est-à-dire si les seuils de temps sont respectés ou non. Si ce n'est pas le cas, nous changeons les données afin d'accepter certaines violations de seuil et nous réappliquons l'heuristique de manière récursive afin d'améliorer la solution (voir figure 2).

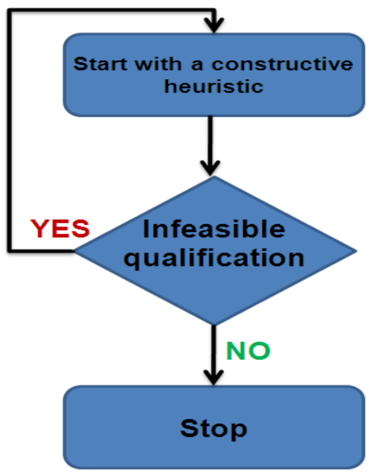


Figure 2: Algorithme récursif

- Si la solution obtenue par une heuristique constructive admet une perte ou des pertes de qualifications de la machine, alors
  - Tant qu'une combinaison de déqualifications n'est pas testée:
    - ∗ Déqualifier les machines avec la combinaison du schéma de déqualification courant,
    - ∗ Appliquer une heuristique constructive avec un nouveau schéma de qualifications :
      - · Si la solution est réalisable, alors accepter la solution
      - · Sinon, retourner la même solution de l'heuristique constructive.
- Sinon, retourner la même solution de l'heuristique constructive.

Metaheuristique : Recuit Simulé (RS)

Le recuit simulé est une métaheuristique inspirée d'un processus utilisé en métallurgie Kirkpatrick et al. (1983). On alterne dans cette dernière des cycles de refroidissement lent et de réchauffage (recuit) qui ont pour effet de minimiser l'énergie du matériau.

- Pour le problème étudié, les trois principaux paramètres critiques du RS sont initialisés de la manière suivante.
  - T_o: La température initiale (ou alternativement la probabilité d'accepter P_o). La valeur initiale de la température est importante pour la performance du RS. Si elle est fixée trop haute, l'algorithme pourrait passer trop de temps sur des solutions de mauvaise qualité. Si elle est fixée trop basse, l'algorithme ne fournira pas de meilleures solutions que les heuristiques constructives. Nous considérons $T_o = 20000$.
  - $\alpha < 1$: Le cefficient de refroidissement. Le facteur dans notre cas est 0.95.
  - t: Décrit la façon dont la température est réduite tout au long de l'algorithme. Nous avons considéré un refroidissement défini par $T(t) = \alpha T(t-1)$. Il permet de réduire la température très rapidement.
- Critère d'arrêt fixe
  - N_{stop}: Le nombre maximal d'itérations sans amélioration. L'algorithme de RS s'arrête soit quand le nombre maximal d'itérations est atteint soit quand le temps d'exécution est de 600 secondes.
- Un voisinage $S(s)$. Tout d'abord, il est important de mentionner que dans les problèmes étudiés, l'espace de solution se compose au maximum de toutes les permutations des jobs, c'est-à-dire $n!$. L'algorithme de RS tente de minimiser l'objectif de PTC et maximiser celui de $PEHF$. Les mouvements se font de permutation en permutation. Nous appelons un voisin de s, une permutation qui est accessible par un mouvement. La façon dont sont générés les voisins a un effet sur l'efficacité du RS. Deux façons de faire des mouvements sont proposées :
  - Insertion "'intra" signifie qu'un job sur une machine donnée est sélectionné dans la position j et inséré avant un autre job à la position i sur la même machine. Chaque fois qu'une insertion "intra" est effectuée, la faisabilité de la séquence obtenue est testée.
  - Insertion "inter" signifie que les jobs sont échangés entre différentes machines.

Le schéma de RS se résume comme suit.

- Déterminer une solution initiale $s \in S$

```
• Selectionner la température initiale $T_o$
• *Répéter*
• $Counter = 0$
    – *Repeat*
    – Generer une solution $s' \in S(s)$
    – Calculer $\delta_{s,s'} = Cost(s') - Cost(s)$
        ∗ Si $\delta_{s,s'} \leq 0$, alors $s = s'$.
        ∗ Sinon, $s = s'$ avec la probabilité $\exp(-\delta_{s,s'}/T(t))$
    – Counter = Counter + 1
    – $T(t) = \alpha T(t-1)$
    – *Jusqu'à* Counter = $M_t$
• t = t + 1
• *Jusqu'à* critère d'arrêt ($N_{stop}$)
```

Résultats numériques

Dans cette section, nous présentons les résultats numériques obtenus avec les heuristiques sur les instances générées. Les résultats montrent, comme prévu, que SCH donne généralement de meilleurs résultats pour la somme des dates de fin, que QCH fournit de meilleures solutions sur le nombre de pertes de qualification machine et que l'heuristique YCH est plus performant sur la maximisation du rendement attendu. L'algorithme récursif et le recuit simulé montrent une performance remarquable dans l'amélioration des solutions (les résultats détaillés se trouvent dans la version complète de la thèse). Les résultats obtenus par les méthodes développées sont comparés avec les solutions exactes obtenues par le solveur standard XPRESS-MP.

0.8 Conclusion générale

Dans cette thèse, nous avons examiné différentes possibilités d'intégration des décisions d'ordonnancement avec des informations et des contraintes issues du contrôle avancé des procédés (*Advanced Process Control-APC*) dans la fabrication de semi-conducteurs. Des discussions et des questions ont été posées sur l'intégration des décisions d'ordonnancement et de l'APC. Nous avons apporté quelques réponses à des questions posées en développant

des idées à travers la définition de nouveaux problèmes d'ordonnancement : *Problème d'ordonnancement avec des contraintes de temps (Problème with Time Constraints-PTC)* et *Problème d'ordonnancement avec l'état de santé des équipements (Problem with Equipment Health Factor-PEHF)*. *PTC* et *PEHF* ont des objectifs multicritères. *PTC* est un problème d'ordonnancement des familles de jobs sur des machines parallèles non identiques en tenant compte des temps de setup et des contraintes de seuil de temps. L'objectif est de planifier les familles de jobs sur les machines en minimisant la somme des dates de fin et les pertes de qualification des machines. La complexité de ce problème réside dans le fait que ce n'est pas seulement une décision d'affectation de jobs sur une machine, mais aussi de sélection de la famille appropriée afin de maintenir les qualifications. Nous avons démontré que ce problème est NP-difficile. Le problème *PEHF* est une extension de *PTC*. Il consiste à ordonnancer des familles de jobs sur des machines parallèles non identiques avec des contraintes de temps et en intégrant la "santé" de l'équipement (*EHF*). L'objectif est d'ordonnancer les familles de jobs différents sur les machines tout en minimisant la somme des dates de fin, les pertes de qualification des machines et en optimisant le rendement attendu qui résulte de l'affectation d'une machine à un job. Ce rendement est défini comme une fonction de *EHF* et de la criticité des jobs.
Pour résoudre ces problèmes des programmes linéaires en nombre entiers, des heuristiques et méta-heuristique ont été proposées, testées et analysées.

0.9 Perspectives

Des pistes de recherche peuvent être proposées concernant la résolution et la modélisation des problèmes définis dans cette cette thèse. Mais avant, il nous semble important d'aller un peu plus loin dans l'étude de la complexité des problèmes définis dans cette thèse. En effet, nous avons pu montrer que les problèmes sont NP-difficile sans pouvoir montrer si leur complexité est au sens faible ou au sens fort. Une réflexion dans ce sens complèterait cette étude complexité.

J'ai proposé dans cette thèse des méthodes exactes (programmation linéaire) pour *PTC* et *PEHF* qui donnent des solutions pour des tailles d'instances relativement petites. Il paraît important d'investiguer d'autres approches pour de plus grandes instances. La génération de colonnes nous paraît être une première approche sérieuse.

Enfin, d'autres dimensions peuvent être intégrées au contrôle avancé des procédés et à l'ordonnancement. Par exemple, certains nouveaux composants dans la conception de produit exigent de vérifier en temps réel que les machines soient qualifiées. Pour s'assurer que la qualification est correctement faite et précise, des wafers tests (*Send Ahead Wafers-SAW*) sont envoyés en éclaireur pour que les machines fonctionnent dans les spécifications voulues. Il s'agit ici, d'intégrer l'ordonnancement des SAW dans le processus d'ordonnancement global tout en minimisant le nombre de SAW qui coûtent chers.

General Introduction

Semiconductor industry is one of the most complex industries. The very large number of processes and the variety in their types are major sources of complexity. In this industry, the wafer is the principle element. A wafer undergoes several hundreds operations. This is due to the fact that processes in semiconductor manufacturing facilities are of a reentrant type, i.e. wafers visit the same machines many times. These environment characteristics and many other aspects, make scheduling in such industries, a complex issue.

Fabrication processes of semiconductors are very precise and require a high level of accuracy. Reliable equipment are required and right recipe parameters should be provided. Advanced Process Control (APC) systems ensure that each process is done following predefined specifications and that each equipment is reliable to process different product types.

In fact, both scheduling and Advanced Process Control (APC) aim at improving the overall fabrication efficiency, and more precisely to maximize the throughput and to minimize the manufacturing cost. There are various types of scheduling strategies and techniques which are used in those industries. Nevertheless, scheduling in semiconductor manufacturing facilities is even more complicated. The difficulty comes from the complex environment and the related high cost of machines and their corresponding maintenance requirements. Many scheduling techniques in semiconductor manufacturing were studied in the literature and research is still active in this domain.

Advanced Process Control has become an indispensable element in the fabrication process with the aggressive competition for better quality and higher productivity. APC utilizes wide variety of techniques in order to keep processes at their specification levels and to survey tools for existing and/or possible defects. To accomplish such goals, statistical techniques, data mining, control charts and control loops are used. The use of such techniques showed a noticeable improvement in throughput, on Overall Equipment Efficiency (OEE), and on the whole manufacturing productivity. Furthermore, many research topics in this domain (APC) are being under study. Modern techniques like Virtual Metrology (VM) are being analyzed and studied.

Most of existing semiconductor fabs use scheduling systems which have no relation with process control systems and *vice-versa*. The objective of this research work concerns the integration of two systems: Scheduling and Advanced Process Control. Here are some questions (not exhaustive related to this integration): What are the benefits of a collaboration between these two systems? What are the challenging problems and the solution approaches that may be proposed? And thereafter, how can we integrate scheduling and APC issues? In this thesis, we try to answer these questions.

Reading Plan This thesis is composed of six chapters. Below, the contents of each chapter are summarized.

- Chapter 1 provides an overview on processes, scheduling and Advanced Process Control in semiconductor manufacturing, where a description of semiconductor manufacturing processes is given. Also, scheduling issues and problems are outlined and a summary on Advanced Process Control components and techniques is provided.

- Chapter 2 addresses the contributions done in the integration of scheduling and Advanced Process Control. In this chapter, different integration issues, possibilities and perspectives are discussed and major integration problems are proposed. Two of the proposed problems, Problem with Time Constraints (PTC) and Problem with Equipment Health Factors ($PEHF$) are considered, studied and analyzed in this thesis.

- Chapter 3 presents a literature review concerning the selected problems (PTC and $PEHF$). Complexity is addressed. Time indexed mixed integer linear programming models are proposed.

- Chapter 4 reports numerical results on the mathematical programming models. Objective functions are considered as a weighted sum of different criteria. These weights are defined in preference vectors. Different types of preference vectors are studied and analyzed. A lexicographical order of criteria is considered. Moreover, an example of the ϵ-constraint method is also provided. A sensitivity study on the Time Constraint (*threshold*) is also done. The results showed the strong compromise between the different criteria of the objective functions for both problems. Numerical experiments showed limitations on the number of jobs, machines and families that a standard solver can handle, which led to developing approximate solution approaches.

- Chapter 5 addresses several constructive heuristics and a metaheuristic (Simulated Annealing, SA) to solve large instances. The performance of dedicated heuristics, a recursive algorithm and SA depends on the considered objectives.

- The last chapter concludes this thesis and various perspectives are proposed and discussed.

Chapter 1

Semiconductor Manufacturing: Processes, Scheduling and Advanced Process Control

1.1 Introduction

In this chapter, we give an overview on the fabrication processes and their different procedures in semiconductor manufacturing. We also address some important scheduling issues. A general overview about Advanced Process Control (APC) is also given, which explores its different components.

1.2 The manufacturing process

The manufacturing process is decomposed in three main types of processes. Section 1.2.1 covers pre-processing processes. Section 1.2.2 covers front-end processes and Section 1.2.3 covers the back-end processes.

1.2.1 Pre-processing processes

Pre-processing processes include (May and Sze (2003)): Silicon polycrystalline growth that results in a cylindrical ingot, Wafer slicing and polishing and Epitaxial silicon deposition that is used to grow a layer of single crystal silicon onto the wafers. The resulting wafers can be used to start the fabrication process of chips which usually includes two stages: Front-end and back-end processes.

Silicon polycrystalline growth

The most commonly used technique for silicon crystal growth is the Czochralski process. A silicon seed is slowly drawn from a crucible of molten silicon to produce a cylindrical ingot of 100 to 300 mm in diameter and up to one meter in length.

Wafer slicing and polishing

Ingots are sliced into individual wafers with a precision thin-bladed saw designed to minimize waste, but rigid enough to cut flatly.

Epitaxial silicon deposition

This is a process used to grow a layer of single crystal silicon from vapor onto a single crystal silicon substrate at high temperatures. Such layers are important to assure device isolation and to avoid junction leakage.

1.2.2 Front-end processes

In the next chapters, we will mostly be interested in front-end processes, where the manufacturing environment forms a challenging issue from both scheduling and APC points of view. Front-end processes include: Photo-resist processes, Photolithography, Etching, Ion implantation and Chemical Mechanical Planarization (CMP).

Photo-resist process

A photo-resist or resist is a photo-sensitive material applied to the wafer in a liquid state in small quantities. The wafer is spun typically at 3000 rounds per minute to spread the material into a uniform layer around 2 micrometers thick.

Photolithography

In this process, a mask is projected, normally using an ultra-violet radiation, on the surface of the wafer to specify the different silicon areas which are involved on a specific layer of the wafer.

Etching

The wafer with patterned photo-resist is then put into an oxide etch process to remove the oxide where there is no pattern. Etching selectively removes portions of semiconductor layers to leave micro-structures on a device.

Ion implantation

In implantation, the dopant molecules are vertically implanted into the surface of the silicon by exposing it to a high-energy ion beam. These molecules aim to change the electrical characteristics of the target area. This is done in order to finally get the different electronic components.

Chemical Mechanical Planarization (CMP)

It is a process used for polishing the surface of the wafer. It can be performed on both oxides and metals. It involves the use of chemical slurries and a circular mechanical action to polish the surface of the wafer.

Figure 1.1 summarizes the processes described above and their sequential order. It also shows the re-entrant type of these processes. For example, a product which already underwent a lithography process and has finished a chemical mechanical polishing process, may need to pass through lithography multiple other times (more than 30 times). These processing steps may be repeated several times, on the same product to separate different layers. This leads to the complex re-entrant nature of such process types, and it is reflected on the scheduling issues in semiconductor manufacturing.

1.2.3 Back-end processes

After front-end processes, wafers are then tested, cut and packaged. This is called the back-end stage. Back-end processes include two major finalizing processes: 1- Testing and 2 - Packaging and assembly.

Testing

Testing includes generally some typical measurements applied to the wafers in order to check its consistency with the predefined specifications. Such measurements typically include wafer flatness, film thickness, electrical properties, critical dimensions. This adds to the complexity of semiconductor manufacturing where sophisticated equipment and qualified workers are necessary.

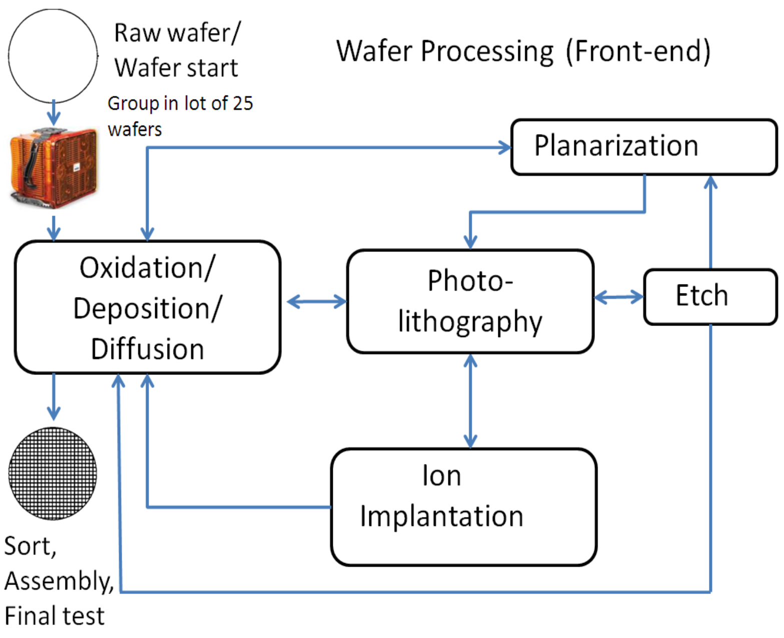


Figure 1.1: Front-end processes and the re-entrant flow (Dauzère-Pérès (2011)).

Packaging and assembly

In order to be connected to other devices or plugged into an electronic card, chips need to be wired. Chips are wired to their appropriate packaging boxes according to their types. A molten plastic material is poured on the whole assembly to form the different well-known existing chips.

1.3 Scheduling

In this section, scheduling in semiconductor manufacturing is addressed. In Section 1.3.1, a definition of scheduling is provided. Scheduling environment in semiconductor manufacturing and the sources of its complexity are discussed in Sections 1.3.2 and 1.3.3. Job shop, batch and cluster tools scheduling are discussed briefly in Sections 1.3.4, 1.3.5 and 1.3.6 respectively.

1.3.1 Definition

Scheduling deals with the allocation of scarce resources to tasks over time. It is a decision making process with the goal of optimizing one or more objectives that satisfy a set of constraints. In all scheduling problems, the tasks (also named jobs), the constraints, the resources and the objective function are the main components. The tasks must be scheduled to optimize a specific objective, and it is often more realistic in practice to consider several criteria (Carlier and Chrétienne (1988)).

1.3.2 Scheduling environment

Scheduling is important for manufacturing as it has major impact on the productivity. This assertion is even more true in Semiconductor Manufacturing (SM) because of the high competition between semiconductor manufacturers, which requires a continuous enhancement of productivity. The timely completion of orders is a high priority, which could be achieved with proper scheduling methodologies. Moreover, cycle time (defined as the time since a machine is assigned to process a job to the time of finishing this job) reduction is usually considered as a primary objective, strongly motivated by high inventory costs.

In semiconductor manufacturing, wafers are usually grouped in lots of 25 wafers. Lots visit the same workshops repeatedly, usually requiring different processing times at different stages. Depending on the tool, the lot, and the processing stage, different setups may be required for the tools. In the sequel, a lot is considered as a job and *vice-versa*.

SM typically involves numerous batch processing operations *e.g.* oxidation, diffusion, deposition, etching, e-beam writing and heat-treatment of wafer fabrication, baking of wafer probing and burn-in of device testing. These operations play an important role in determining how the system performs in terms of throughput, inventories and cycle time. They generally add variability to the system, because the items wait to form a batch and, upon service completion, are released to downstream operations (Fowler et al. (2002)). Further, effective scheduling of many batch processors in SM system is important in terms of improving the due date compliance of the whole system. Also, it is important for significant reduction of the overall cycle time due to the long processing times compared to other manufacturing processes. The environment in semiconductor manufacturing facilities, also called wafer fabs, can be seen as a highly complex job-shop, involving multiple types of

work centers, large and changing varieties of products, sequence dependent setup times, reentrant process flows, etc. This problem has been investigated from a variety of perspectives resulting in several analytical techniques combining generic as well as problem-specific strategies.
A vast amount of literature exists on job shop scheduling due to the continuous and improved research efforts in this field over the last five to six decades. Most of the batch scheduling approaches to the semiconductor manufacturing scheduling problems can be classified into four categories: Heuristic rules, mathematical programming techniques, neighborhood search methods, and artificial intelligence techniques. Figure 1.2, taken from Moench et al. (2011) summarizes deterministic scheduling in wafer fabs. It shows the various machine environments, process restrictions and objectives that can be found in a semiconductor fab.

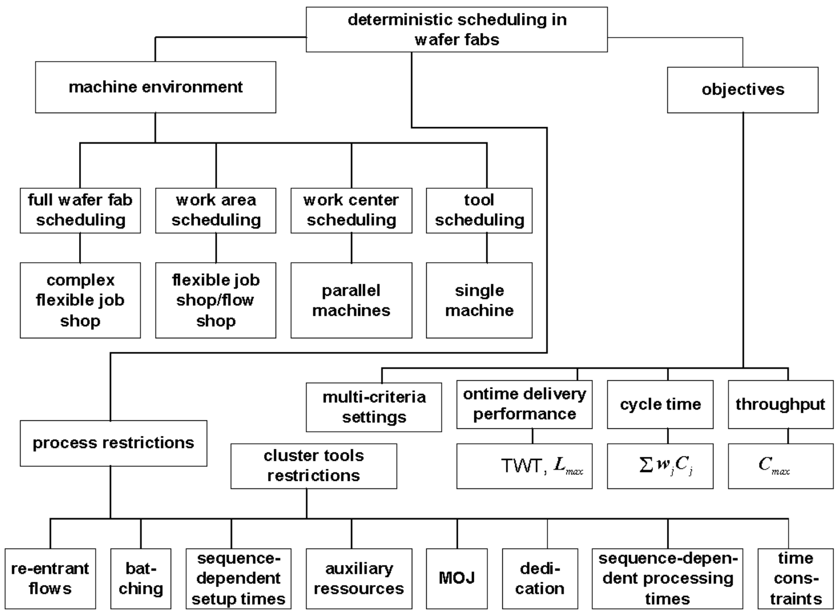


Figure 1.2: Deterministic scheduling in wafer fabs (Moench et al. (2011)).

1.3.3 Sources of scheduling complexity

Wafer fabrication routes consist of 300 to 500 processing steps. There are a number of unique features/requirements in these steps that make the problem of optimally scheduling operations a complex issue. These include:

- Re-entrant flows: wafers revisit the same workshop at different stages of their process flow, so that lots at different stages of their manufacturing cycle compete with each other for resources.

- Non-Uniform Load: due to the diversity in the production system, cycle times vary by tool and process step. In addition, tools can process wafers in lots, batches or process single wafers. Thus, machines can off-load multiple lots onto tools that are capable of processing only one lot at a time. This results in non-linear loading or flow of products in the factory.
- Setup Times: some tools have significant sequence-dependent setup times that need to be considered when computing schedules.
- Downtimes: the manufacturing system can have random tool failures.
- Rework: during the course of processing, some wafers in a lot may need to be reworked, after which they regroup in the lot. Reworked wafers need to be prioritized as the lot will wait before proceeding to the next processing step. Some wafer lots may need to be rejected because of defects.
- Hot Lots: during processing, it may happen that some lots are given higher priority compared to other lots in the factory to meet delivery due dates or other processing requirements.
- Auxiliary Resources: at different processing steps in wafer fabrication, auxiliary resources are required (for example, reticles or masks in photolithography). These auxiliary resources are often limited in number.
- Test/Monitor Wafers: along with production lots, there might be some test or monitor wafers in the systems which are being used for testing/experimenting new products. Also, there might be a requirement that the system cannot handle more than a certain number of test wafers or needs a minimum number of these wafers.
- Send Ahead Wafers: it may be required that some wafers of a lot are sent for processing after which they are inspected. Based on the outcome of the inspection the remaining wafers of the lot are processed and merged back into same lot.
- Processing Time Windows: in certain processing steps in the flow, *e.g.* a batch processing furnace operation following a clean or etch operation, it may be required that the next operations carried out within a certain time window after the first operation.

- Rules/Constraints for Scheduling: each step in the process flow might have rules/constraints that govern the way products and processes are scheduled, and these rules/constraints might vary from one fab to another.

- Machine qualifications: a machine must be qualified to process an operation, i.e. some parameters have to be set or adjusted (*e.g.* temperature, pressure, etc) to pre-defined specifications. This notion is particularly important in high-mix product environment.

A fab scheduling system must schedule lots of wafers taking all the above listed requirements into consideration. The system should also be able to reschedule lots, in real time, to accommodate changes in the manufacturing environment. This involves exploration of a huge solution space to find the best schedule or solution, which meets all constraints and minimizes the cost. And, in order to be useful in practice, the scheduling and rescheduling steps need to be done rapidly.

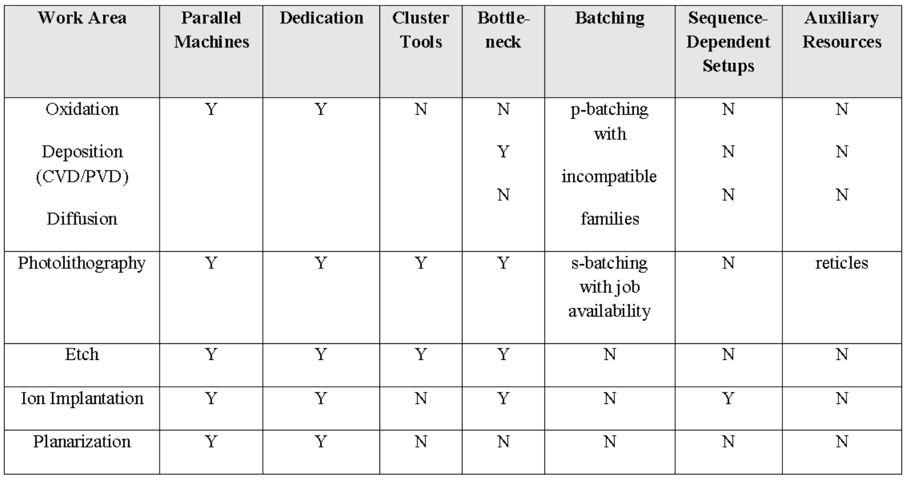
1.3.4 Job shop scheduling

A wafer fab can be viewed as a highly complex job shop containing a number of single-server and/or multi-server stations (Wein (1988)). A vast amount of literature exists on job shop scheduling because of the continuous and improved research efforts in this field over the last six decades. The research focus on scheduling in semiconductor manufacturing has only come into existence in the last 20 years. It has become a very important issue nowadays for the overall growth of the economy, since the semiconductor industry has seen a phase of rapid advancement during this period. A review on job-shop scheduling techniques in semiconductor manufacturing was presented in Gupta and Sivakumar (2006). In this paper, the authors provided a list of scheduling techniques that are used in semiconductor manufacturing.

Dispatching heuristics provide schedules quickly, but there is no guarantee of optimality using these rules. Mathematical programming techniques can provide optimal solutions, but computational times are prohibitive for large problem sets. Neighborhood search methods are capable of finding very good solutions in a reasonable amount of computational time. The choice of a scheduling technique depends on several parameters that come from the type of the problem as well as the needs and priorities.

1.3.5 Batch scheduling

Batching jobs is a very common policy in the manufacturing system of most of the industries. The main reasons for batching are avoidance of set ups and/or facilitation of material handling. We define the batch as the set of jobs that are processed together. We call the number of jobs in a processing cycle the batch size.
Figure 1.3 taken from Moench et al. (2011) summarizes the characteristics of scheduling problems found in different work areas of wafer fabs. However, in this thesis, we are more interested in problems that are found in photolithography area. Further, We note that time constraints were not considered because they are often not only related to a single work area but across different work areas.



Work Area	Parallel Machines	Dedication	Cluster Tools	Bottle-neck	Batching	Sequence-Dependent Setups	Auxiliary Resources
Oxidation	Y	Y	N	N	p-batching with incompatible families	N	N
Deposition (CVD/PVD)				Y		N	N
Diffusion				N		N	N
Photolithography	Y	Y	Y	Y	s-batching with job availability	N	reticles
Etch	Y	Y	Y	Y	N	N	N
Ion Implantation	Y	Y	N	Y	N	Y	N
Planarization	Y	Y	N	N	N	N	N

Figure 1.3: Characteristics of scheduling problems in wafer fabs (Moench et al. (2011)).

A review on Scheduling of Batch Processors (SBP) problems in semiconductor manufacturing can be found in (Mathirajan and Sivakumar (2006)). The authors classify SBP problems in SM into several groups. The classification results are presented based on various distributions, and different methodologies applied for SBP problems in SM are briefly highlighted. Batching occurs in two versions: Serial batching and parallel batching. In serial batching, jobs may be batched if they share the same setup on a machine and one job is processed at a time. In parallel batching, several jobs can be processed simultaneously as a batch. A good analysis of both types of scheduling with batching was provided in (Potts and Wassenhove (1992)), (Albers (1993)), (Webster and Baker (1995)), (Brucker et al. (1998)) and (Potts and Kovalyov (2000)). When the batch size becomes very large, the problem of uniformity of machines involved in processing this batch appears. Hence, cluster tools are used to minimize the difference in uniformity. In Section 1.3.6, we briefly discuss cluster tools.

1.3.6 Cluster Tools

Cluster tools, which combine several single-wafer processing modules with wafer handling robots in a closed environment, have been increasingly used in wafer fabrication processes. We recall that the basic material handling unit in a fab is a wafer lot that consists of 25 wafers. However, as the wafer size increases and quality requirements become stricter due to circuit shrinkage, the batch processing technology becomes difficult to ensure wafer quality. This is because it is hard to control uniformity of gas or chemical diffusion on multiple large wafer surfaces. Consequently, cluster tools or track equipment have been increasingly used, including photolithography, etching, deposition, and even testing (Lee (2008)).

1.4 Advanced Process Control (APC)

1.4.1 Introduction

The basic concept of equipment and process control lies in Advanced Equipment Control (AEC) and/or Advanced Process Control systems, which have become vital in any type of industries and especially in semiconductor manufacturing. Advanced Process Control (APC) has become an essential component for factory control to maintain the continuous improvement of device yield and reliability at reduced cost. Generally, APC enables cost reduction in manufacturing through the following improvements:

- faster process development,
- reduction of monitor wafers,
- decrease in process variation, increase of yield, and shorter control loops in case of failure or drift,
- more stable processes ensure product reliability,
- increased equipment utilization.

In semiconductor industry, it is very important to improve throughput, yield and performance of products. APC is widely used to increase stability and accuracy in fabrication. It is made of SPC (Statistical Process Control), FDC (Fault Detection and Classification), R2R (Run-to-Run) and more recently VM (Virtual Metrology). SPC and FDC focus on detection of equipment faults or process abnormality. Feedback APC techniques such as Model Based Process Control (MBPC) and Exponentially Weighted Moving Average (EWMA) are used to reduce systematic error and VM for predicting metrological information at wafer level. These basic components of APC are introduced in the following sections.

1.4.2 Statistical Process Control (SPC)

Statistical process control (SPC) involves the use of statistical techniques to measure and analyze process variations. Most often used for manufacturing processes, the intent of SPC is to monitor product quality and check that processes are within fixed targets. SPC is used to monitor the consistency of processes used to manufacture a product as designed. It aims at keeping processes under control. No matter how good or bad the design is, SPC can ensure that the product is being manufactured as designed and intended. Thus, SPC will not improve the reliability of a poorly designed product, but can be used to maintain the consistency of how the product is made and therefore, of the manufactured product itself and its as-designed reliability (Montgomery (2004)).

A primary tool used for SPC is the control chart, i.e. a graphical representation of certain descriptive statistics for specific quantitative measurements of the manufacturing process. These descriptive statistics are displayed in the control chart in comparison to their *in-control* sampling distributions. The comparison detects any unusual variation in a manufacturing process, which could indicate a problem with the process. Several different descriptive statistics can be used in control charts. There are many types of control charts that can test for different causes. Some of the basic tasks, which are associated with SPC, include: Surveillance and feedback, signaling a problem with the process, detection of assignable causes of variation, reducing need for inspection, monitoring of process quality and providing mechanism to make process changes and track effects of those changes.

Tools used in SPC

SPC can be applied to any process. Its seven major tools are (Montgomery (2004)): Histogram or stem-and-leaf plot, check sheet, Pareto chart, Cause-and-effect diagram, defect and concentration diagram, scatter diagram, control chart and process Capability. Although these tools, often called *the magnificent seven*, are an important part of SPC, they are usually used to complement each other, rather than employed as stand-alone techniques. In what follows, control charts and process capability are discussed. These tools were chosen among all others for the sake of providing a detailed and relevant example.

Control charts

Control charts are used to detect whether a process is statistically stable, which means that the process is under statistical control. The procedures behind the application of control charts would be explained by the following scenario: a process is sampled regularly, and the collected data is plotted which represents some measure of performance such as mean, range, number of defects, or any other variable. The process can then be checked graphically, for example, whether it is under statistical control or not. Otherwise, certain predefined action corresponding to the type of extrusion we obtained, should be taken. For further details, the reader can refer to Montgomery (2004).

Process capability index

The process capability index expresses the variability in process characteristics, i.e. whether the process is capable of producing products which conforms to specifications or not. Process capability studies distinguish between conformance to control limits and conformance to specification limits which are also called tolerance limits. For example, if the process mean lies within control limits, then eventually all points will remain within control limits.

Process capability is also another important concept in SPC. It also refers to the process uniformity. The variability of a process characteristic is a measure of the uniformity of a single process variable. The capability of a process is determined only when common causes of variability are present. Determination of the capability of a process requires the estimation of mean and standard deviation of a variable characteristic of the process (Montgomery (2004)).

1.4.3 Fault Detection and Classification (FDC)

The precise control of equipment status and performance is essential in modern semiconductor manufacturing facilities. The idea is that, by having well controlled internal equipment and process parameters, higher yields are expected. FDC detects abnormalities of such key parameters within a relatively short time, which allow the related equipment to be stopped, and scrap to be avoided. The main purpose of FDC is at first to detect an abnormal status of the equipment or the process running on it (Fault Detection - FD). The second step is then to classify the detected failure, for instance a leak in the chamber or a problem in a power supply, and to give engineers and technicians a hint where to start to search for the root cause (Fault Classification - FC).
A well implemented FDC system across a fab will provide many advantages on the whole manufacturing process. At the level of equipment monitoring for example, prevention of accident, enhancement of throughput, reduction of test wafer, enhanced Overall Equipment Efficiency (OEE), real time monitoring, preventive maintenance, and yield enhancement by reduction of scraps and thus minimizing the cost, are some of these benefits.
A typical FDC system collects on-line a large amount of data from processes by equipment sensors for every process run. They are called process variables or FDC data. Some reliable available FDC data are essential to construct a decision rule able to detect as quickly as possible an abnormal evolution of the system (for example a machine) in order to prevent more critical problems in the future (He and Wang (2007)).
Moreover, fault diagnosis and classification begin with the selection of a measurement plan, which is the data acquisition approach used to derive a symptom vector. The selection of an appropriate set of measurements whose deviations from expectations are used for diagnosis is critical. The cost, speed, and accuracy of data acquisition are obviously important, since even a set of accurate but poorly selected measurements can be of limited use.
Further, maintenance diagnosis is another technique which aims at avoiding potential component failures based on historical performance. The available data consist of the number of failures that a given component has experienced as well as the component age. Usually a neural network approach is being proposed and developed for the implementation of maintenance diagnosis (Kim and May (1997)).

1.4.4 Run to Run (R2R) control

Run-to-run control is a form of discrete process and machine control in which the product recipe with respect to a particular machine process is modified ex situ, i.e. between machine "run", so as to minimize process drift, shift, and variability (Moyne et al. (2000)).
This type of control is event-driven, where the events include the determination and reporting of pre- and/or post process *ex situ* metrology data, and the requirement of the tool to begin processing. The control could be on a wafer-to-wafer, lot-to-lot or batch-to-batch basis. The metrology and automation scheme for R2R control can vary widely (Jedidi (2009)). For example, the metrology is generally limited to *ex situ* metrology, but could include *in situ* equipment state.

Types of R2R systems

A wide range of R2R systems currently exist in semiconductor manufacturing. They can be categorized into three basic categories:

- Post-process quality measurement data, where the measurement data may be in the form of ex situ post-process metrology (traditional), but could also include in situ data compiled during the process. Knowing that for most cases, the data may or may not be available for every wafer, batch, or control event. Moreover, it is not trivial to have the pre-process measurement data available all the time.
- A dynamic model of the process is maintained in the controller that relates the post-process quality data to tunable process "recipe" inputs. Using this model, the controller is able to provide "suggestions" for process improvement as necessary based on post-process quality data values. The model is dynamic in that it attempts to track drifts in equipment and process quality parameters on a run-to-run basis.
- Process improvement control actions, i.e. process input parameter adjustments are set once during each "run" based on suggestions by the controller.

Figure 1.4 shows the feed-forward and feed-backward principles in a $R2R$ controller. The metrology results before a production process are sent to update process parameters, and hence the feed-forward. On the other hand, post process metrology results are sent in the backward direction which corresponds to feed-backward.

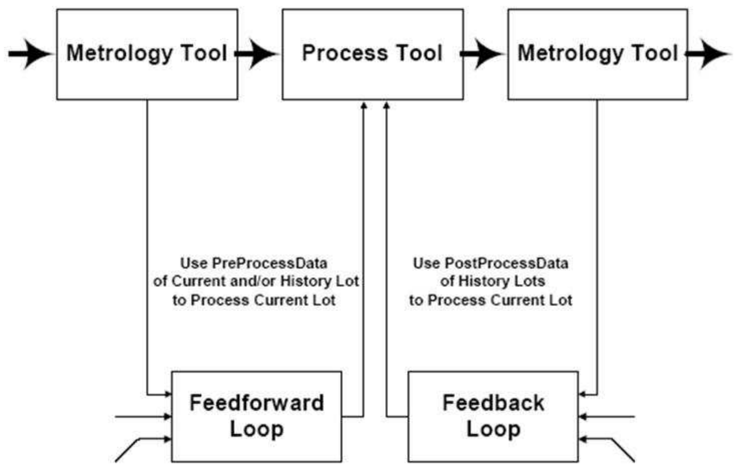

Metrology Tool
Process Tool
Metrology Tool
Use PreProcessData of Current and/or History Lot to Process Current Lot
Use PostProcessData of History Lots to Process Current Lot
Feedforward Loop
Feedback Loop

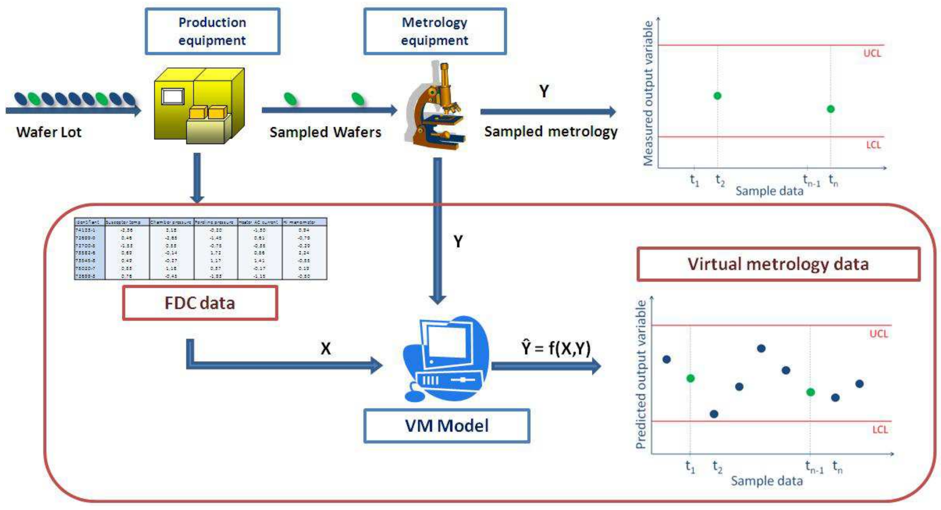
Figure 1.4: Feed-forward and feed-backward in R2R control.

Regarding all these R2R characteristics and issues, the idea of scheduling with the needs and/or constraints of a R2R system seems to become more realistic and relevant when considering modern scheduling and planning techniques.

1.4.5 Virtual Metrology (VM)

VM refers to the technology that estimates the process results based on the previous metrology measurements and equipment sensor data, instead of performing actual measures. Most APC techniques strongly depend on the physical measurement provided by metrology tools (Qin et al. (2006)). Critical wafer parameters are measured, such as, for example, the thickness and/or the uniformity of thin films. If a wafer is *mis-processed* in an early stage but detected at the wafer acceptance test, unnecessary resource consumption is unavoidable. Measuring the quality of every wafer after each process step could avoid late wafer scraps but it is too expensive and time consuming. Therefore, metrology, as it is employed for product quality monitoring today, can only cover a small fraction of sampled wafers.
Virtual Metrology (VM) in contrast enables prediction of every wafer metrology measurement based on production equipment FDC data and previous metrology results (Chang et al. (2006)), (Chen et al. (2005)), and (Cheng et al. (2007)). This is achieved by defining and applying predictive models for metrology outputs (physical measurements) as a function of metrology

and equipment data of current and previous steps of fabrication (Hung et al. (2007)), (Khan et al. (2007)) and (Lin et al. (2006)).
Of course it is necessary to collect data from equipment sensors to characterize physical and chemical reactions process chambers. Consequently, reliable and accurate FDC data are essential in VM models (Su et al. (2008)). The objective of a VM module is to develop a robust prediction that can estimate metrology and which is able to handle process drifts whether they are induced by preventive maintenance actions or not.
Figure 1.5 presents an illustrative diagram of virtual metrology. A step of stand-alone metrology with a sampling of two wafers measured by lot is considered. With only these two measurements and the history of FDC parameters of this lot, we can build a mathematical model to predict the values for all the other wafers which is a very challenging goal to accomplish.



Identifiant	Susceptor temp	Chamber pressure	Pending pressure	Heater AC current	Hi manometer
74125-1	-2,56	2,15	-0,20	-1,50	0,54
72669-0	0,46	-2,65	-1,46	0,61	-0,75
72700-5	-1,55	0,55	-0,75	-0,56	-0,29
75562-6	0,69	-0,14	1,72	0,66	2,24
75545-5	0,49	-0,27	1,17	1,41	-0,55
75020-7	0,55	1,16	0,57	-0,17	0,19
72669-5	0,76	-0,45	-1,95	-1,15	-0,50

Figure 1.5: Virtual metrology.

Virtual metrology can enable wafer to wafer control without additional real metrology. It is a result from a real need in modern fabs.

Advantages of VM

There are many advantages to virtual metrology including:

- **Reducing wafer scraps:** Process inspection can be performed through VM for every wafer to sustain yield performance.

- **Tighter process control:** VM provides a basis to overcome the metrology delay problem for run-to-run control.
- **Increasing throughput:** Wafer handling from process tool to metrology tool can be reduced and, thus, production cycle time can be shortened.

The development of virtual metrology does not aim to replace stand-alone metrology tools, but to assist in achieving total quality management and process control. However, the research progress is still tardy because of the lack of relevant theory. While implementing VM, various needs come out as for example the data collected from metrology tasks. These needs can be seen as constraints for a scheduler, and hence scheduling while respecting such needs becomes more challenging especially when they are considered as hard constraints.

1.5 Conclusion

Based on this chapter, we can now imagine new integration problems and propose new methods to improve production control and efficiency in semiconductor manufacturing. A factory integration strategy is described by the International Technology Roadmap for Semiconductors (ITRS) which mentions the integration of all factory systems to have at the end a fully integrated semiconductor industry in all its systems. Alternatively, new integration ideas were elaborated by major semiconductor manufacturers such as INTEL, AMD, and STMicroelectronics. We discuss in the next chapter, possible integration ideas and we define new problems that come out when merging APC and scheduling systems.

Chapter 2

Integration of Scheduling and Advanced Process Control

2.1 Introduction

In this chapter, we are going to introduce Advanced Process Control (APC). The benefits of APC are: Improved process performance, increased overall equipment efficiency, improved throughput and reduction of Non Product Wafers (NPW) (Zaugg (2008)). Alternatively, the benefits of smart scheduling methods are basically: reduced cycle time and increased resource utilization.
Nowadays, in most semiconductor manufacturing facilities, scheduling decisions are taken depending on more or less advanced lot to tool assignment rules, in which the lot with the highest priority is dispatched on the next available tool. However, considering the existing constraints and decision possibilities, there are other types of constraints and criteria that must be taken into consideration, such as equipment availability, equipment states, time constraints, due dates, bottleneck resources, etc (Anderson and Hanish (2007)). In the past, and still in some current semiconductor fabs, industry system applications are being isolated from each other. This is still valid for scheduling and APC systems. However, decisions in modern semiconductor fabs have become more integrated and are moving towards service-oriented policies. This new philosophy aims at creating a unified platform between the different existing vendors in order to communicate more easily. In this chapter, we discuss in Section 2.2 the interactions between scheduling and APC. We address challenging integration problems in Section 2.3 before concluding in Section 2.4.

2.2 Interactions between scheduling and APC

In a more and more complex economic world, the recession of some countries and competition in many sectors increased. Industries are permanently searching for solutions that allow them to remain competitive. This creates a need for information sharing between different decision systems. The goal is to assure that the decisions taken at one level do not negatively impact decisions taken at other levels. This leads to global and challenging problems with more complex constraints and multiple objectives (often conflicting). In this chapter, we are interested in the integration of scheduling and APC.

2.2.1 Motivations

Some of the major motivations for integrating scheduling and APC decisions in a semiconductor fab are summarized below.

- Operations related to APC (like measurements for instance) interfere with scheduling decisions by modifying processing modes and lists of operations to perform.

- Using real-time machine capabilities, which are available through time and provided by APC methods will help scheduling to improve cycle times and also yield.

- Equipment qualification is crucial, in spite of its high cost value. Thus, scheduling algorithms should decide how operations will be assigned to qualified machines, and balance between the corresponding cost and the quality. In this context, machines may be considered as completely qualified or disqualified, or we may even consider levels in machine qualification (machine eligibility). Moreover, enormous amount of information related to machine and process states are collected by APC equipment, thus an Equipment/process Health Factor (EHF) could be available which is an indicator on the state of equipment (*e.g.* bad, good, excellent). Again, scheduling decisions may become more effective by promoting reliable processes and equipment.

- Parameters of R2R control loops regularly need to be updated. When a machine is used to manufacture different types of products, test wafers must be sent to keep the associated control loops updated. Hence, a scheduling algorithm which takes into consideration such a constraint will be more effective. This constraint may be presented for example as a time constraint related to control system loops, as well as the number of lots that were processed on a given machine. In chapter 3, a new time constraint related to the APC system is presented and is being integrated into a scheduling problem of parallel machines with different eligibility characteristics.

These are some of the major reasons behind the idea of integrating scheduling and APC decisions in any possible type of industry, and especially in semiconductor industry because its complexity, the enormous number of control loops and re-entrant processes. It seems promising to explore the possibilities and problems related to integration in a research avenue that takes as its starting point APC information and aims at integration scheduling conditions and *vice-versa*. In the following section, we detail the interactions between scheduling and APC to later define new integration problems. Indeed, there are two ways of thinking at the interactions between scheduling and APC; both are detailed in the subsequent sections.

- APC information for efficient scheduling (Section 2.2.2).
- Scheduling with APC constraints or requirements (Section 2.2.3).

2.2.2 APC information for efficient scheduling

Data available in the APC control system may provide the scheduling system with effective information. These data are used later as a new criterion under the form of constraints and/or objective, to improve scheduling decisions.
For example, a run-to-run controller detects the polish rate of a Chemical Mechanical Planarization (CMP) tool. We consider the case where we have m CMP machines, each with its own polish rate. A scheduler informed by these different polish rates will be able to send/batch a lot/wafer to the machine with the highest polish rate. This corresponds to the Shortest Processing Time (SPT) dispatching rule. Moreover, an APC controller can basically provide the current process conditions and parameters for certain machines. Thus, if there are m machines of the same type and with almost similar processing conditions, always according to the data provided by APC controllers, and if the dispatching system has this information, it would be able to send batches of lots to the machines as if they were one entity. In addition, if multiple tools were set up to run a given lot, the dispatching system could then decide to send a lot to the process tool with the actual shortest processing time.

Further, the APC system can provide a capability index of a tool/process, also known as the process performance index (C_{pk}). It specifies whether the tool/process is capable of producing a product that conforms to the specifications or not. This information provided by APC system helps the scheduling system to select tools with high C_{pk} level for critical layers or process areas. This may result in an improved yield. In addition, let us recall that the APC control system gives information on the equipment state, which can be used to define a health factor associated to each equipment, and which vary over time (Montgomery (2004)). This factor was recently defined in the literature, and is known as Equipment Health Factor (EHF). Thus, scheduling techniques which take these factors into account, will be more precise and efficient by routing lots or wafers to machines that are more reliable. Real-time information on processes may also be collected such as real exact processing times. These data may also interfere with scheduling decisions.

Nevertheless, tools that are assigned to perform a given process should be qualified before starting this process (equipment/machine qualification). Different from the equipment configuration needed at the beginning of processing a new family type lot/wafer (setup), regular equipment qualifications are frequently fulfilled according to predefined strategies based on the fabs' policy and on the available expertise. This qualification is done usually by sending special types of wafers to test the equipment and/or process (e.g. Send Ahead Wafers, SAW). Here also, we can use this information to develop scheduling techniques with a clearer view on which equipment is qualified to perform which process, and at which cost (e.g. expected yield, probability of loss, etc). Further, smart dynamic sampling for metrology operations is getting more importance. Typically, after a lot is completed on a process tool, it will undergo test operations on a dedicated metrology tool (e.g. the thickness of a silicon dioxide layer). Hence, some of the operations are for processing and the others for testing. The scheduling system should then not only decide how lots are processed, but also if and how they are measured (Dauzère-Pérès et al. (2010)). How to select lots to measure, and how to schedule the selected lots are questions that open the door to many possibilities and proposed solutions. Sampling for metrology is discussed later in this chapter.

Furthermore, an APC-abort is defined to be the inability to start a lot due to the lack of information in the APC system. This forms a disadvantage for the flow of the production process, hence reducing these aborts is an interesting objective. A dispatching system that is informed by APC constraints, will use this knowledge to dispatch lots that would keep the control loops (e.g. R2R) updated. For example, knowing that the parameters of a given control loop on a given machine are going to be outdated, the dispatching system will schedule lots that compensate for these parameters in the considered machine. Thus, the control loop parameters are updated before that they are out of specifications, which usually requires a tool recalibration. Moreover, this helps to decreases cycle times by keeping the machines eligible to process lots.

Equipment Health Factor

Not fully studied and noted in the literature, the Equipment Health Factor (EHF) or Index (EHI) is a rich domain of research, which still needs to be accurately specified. Nevertheless, one can define the health factor of equipment as an indicator of the current state of the equipment depending on several measured parameters such as temperature, pressure, etc (Guo et al. (1998)). This factor may specify for instance the zone within which the machine works with high reliability or, on the contrary, may fail. The goal is how to determine an appropriate function that has as inputs the measured parameters and that gives as output the EHF. Few models and methods were proposed in that field, but actually there is no final exact method to determine a precise factor (Chen and Wu (2007)).

Such a factor may be helpful in scheduling decisions to guide the lots to routes, i.e. tools, which are more reliable. More precisely, consider the case where lots of different family types, with different criticality levels, are to be scheduled to tools with various health factors for each family type. Then, the decision of assigning critical lots/wafers to *healthy* equipment become more challenging while trying to optimize some scheduling criteria such as the makespan or sum of completion times. Scheduling with equipment health factor will be discussed in more details in Section 2.3.

Wafer Quality Indicator

Consequently to what was mentioned above, and despite the fact that it is not trivial to know exactly and instantaneously a machine state, and that research is still required in this field, it is important to mention that, by combining data collected from SPC and FDC control systems, a basic machine health factor could be deduced. This basic indicator/factor will give the scheduler a view on the states of the machines.

Let us now consider the case where a lot arrives on a certain machine m. We suppose that this wafer will be tested. Combining the result of this test with the equipment health factor, an indicator of wafer quality could be deduced. This idea would be presented by a quality level indicator. This indicator could be associated to each wafer, and will be used by the scheduler to decide where to send the associated lot. In other words, the scheduler now knows the states of all the future tools and the state of the lot itself. Thus, it may change the predefined route of the lot depending on both factors: the equipment health factor and the wafer quality level indicator. Hence, fully dynamic scheduling plans would be available instantaneously.

The idea of using dynamic data is hard to apply in practice. This is why we add some restrictions and constraints in order to make it more applicable. Actually, the idea of considering a wafer quality indicator is complementary to the EHF addressed above. A combination of wafer-tool leads to define an associated cost of assignment that could be represented for example as of a production yield and/or a probability of losing a wafer/lot.

Equipment Qualification

Equipment qualification is also an important issue in semiconductor fabs. Qualification can help improving tool efficiency, minimize work-in-process, and reduce cycle time. Machine qualifications management impact the overall performance of a wafer fab (Johnzén (2009), Johnzén et al. (2010)). Typically, lots cannot be sent to a tool unless that tool is qualified. Qualification is usually maintained by sending test wafers (TW) to the tool (Faruqi et al. (2008)). Hence, scheduling decisions are similarly affected by equipment qualification.
The management of equipment qualification can be seen as a tactical problem. Long term equipment qualification decisions are required especially when we know that, for instance, the initial qualification process can take a large amount of time depending on the process nature in the manufacturing chain. Actually, we can consider two levels of qualifications: Equipment qualification level and process condition qualification. To start processing a lot on a tool, both qualifications are required. A scheduler who takes into account qualified and non-qualified tools and processes will effectively decide much better how to route lots in the fab.

2.2.3 Scheduling with APC constraints or requirements

Scheduling systems may enhance the performance of APC control systems in particular knowing that, in such an environment, there are multiple tools, products and processes. As an example, there would be effectively thousands of control loops to be keep updated. More precisely, recipe adjustments done by run-to-run regulation loop require regular metrology operations to perform for each process and product type. Otherwise, the processing conditions are no longer in accordance to specifications and the risk of losing a lot/wafer increases.
To illustrate, let us consider the following scenario: a tool serves several types of products, *e.g.* three type. After an interval of time (threshold), let us assume that it only received lots of two product types and the third has not being processed in the interval.
The regulations in the loop become out of date (if the interval was sufficiently large), so any lot of the third product type will not be accepted. The control loop must be re-calibrated (qualified) for the corresponding product. To maintain qualification, it is then necessary to achieve the processing of a lot of the third product type before the end of the time interval. It is obvious that a scheduling system that takes into account this constraint will improve the Overall Equipment Efficiency (OEE) and decrease the number of test wafers.

In addition, time limits associated to special process types are also difficult to handle. In some process areas, there is a maximum allowed amount of time for lots between process steps. For instance, a material that has been ashed (e.g. removal of a photo-resist using plasma ashing) must go through the subsequent step within a specified time limit, or else the ash procedure should be repeated. Consequently, a scheduling system which takes these constraints into account would decrease redundant steps and consecutively would improve the overall cycle time and fab yield.
In what follows, we address two examples of scheduling with APC constraints or requirements. Dynamic sampling and preventive maintenance were chosen as they are considered as two relevant realistic semiconductor fab issues.

Dynamic sampling

In order to dissipate for process drifts and variations, APC controllers demand a huge amount of measurement data collected from wafers. This is not always easy to be done because metrology tools are rare and expensive resources that do not bring added value to the product. So, it is almost impossible to get all the necessary measurements and similarly for all the

wafers. In the extreme case, if all lots/wafers were measured, then this will result in very large waiting times and number of operations. Alternatively, different machines have various processing times and this result in an increase in cycle time variability. Hence, the idea of dynamic sampling was elaborated, which consists of dynamically deciding which products should be measured to keep certain control parameters of different APC systems (e.g. R2R, VM) updated (Dauzère-Pérès et al. (2010)).
More precisely, many current run-to-run applications use a fixed measurement scheme. The measurements steps are considered as non-added value in terms of the final product, and they have a negative impact on cycle time and throughput. However, insufficient measurements lead to poor controller performance. Information obtained from the control system could be used by a dispatcher/scheduler at the metrology steps to determine whether a lot should be inspected or not. For example, consider the situation where the controller estimates that: the process is performed as expected (in control); has an acceptable process performance index, usually denoted in the literature by C_{pk}; and its (R2R) control loop parameters have been updated recently. Thus, depending on these circumstances, the metrology operation of this lot/wafer could be canceled, this is what we call smart *skipping*. On the other hand, if the controller detects any "out-of-control", then the number of inspected lots is increased and this is equivalent to an increase of the sampling rate.

In semiconductor manufacturing, a sampling strategy determines a rate of measurements based on statistics. These rates can be obtained empirically based on the knowledge on products during their various manufacturing stages, and more effectively by taking into consideration their life cycles (Bousetta and Cross (2005)). The major obstacle is that an increase in production volumes leads to a saturation of metrology tools. Hence, not all lots that should be measured can actually be measured. This implies that there should be a selection decision of the lots to be measured. This decision depends on many factors, for example: lot type, lot priority, metrology capacity, etc. Answering the questions of what, when and how to choose a lot for inspection forms a sampling strategy. Various dynamic sampling strategies were developed. For instance, Lee et al. (2003) outline a strategy to control the queues and to skip certain measurements. In addition, Purdy (2007) presents a sequence modeling strategy for selecting lots for measurement. Further, Holfeld et al. (2007) describe a system applied to an entire manufacturing unit and that balances the needs of metrology, and minimizes the risk under capacity and cycle time constraints.
Moreover, current dynamic sampling techniques are based on statistical no-

tions which determine the sampling rate. However, instantaneous information taken from the APC control system are not yet well exploited. If this was the case, dynamic sampling techniques may become "smarter" (smart sampling). Furthermore, from our point of view, other circumstances may also be taken into account depending on the vast experience in the field of semiconductor manufacturing.
Let us take as an example the following case: Suppose we just started processing a lot on a machine, then it is clear that the results of the measurements of the first wafer/lot is indispensable to calibrate the control loop parameters, while it is not of great importance for the next ones. The difficulty lies in determining the required number of modifications necessary for maximum efficiency and how to modify this number over time.

Preventive Maintenance

Preventive Maintenance (PM) is a crucial activity in semiconductor manufacturing since there are no tools in a fab that are 100% reliable. A good PM schedule can increase the availability of tools by trading off between the planned unproductive downtime versus the risk of unscheduled downtime due to tool failures.
Unscheduled downtimes not only induce the loss of productivity, but also disturb the manufacturing process. Thus, in order to lessen these negative effects, PM tasks have to be scheduled carefully and comprehensively. There is research on the integration of preventive maintenance and scheduling decisions and their inter dependency (Cassady and Kutanoglu (2005)). However, modern semiconductor fabs need more than just preventive maintenance policies; shifting from classical preventive maintenance to predictive maintenance has become a need. Hence, the integration of APC and scheduling in a single predictive preventive maintenance policy will help to decrease equipment downtimes and thus to increase the availability of tools.

APC components, and more specifically SPC and FDC, can provide real-time information on production equipment. The idea is to benefit from this information for the sake of PM. Data available in SPC databases may be useful to predict potential future machine failures by applying statistical techniques. Also, the states of tools available in FDC control systems almost real time may be used to predict possible short-term tools failures. By merging the information on machine states into the scheduling system, we expect that better scheduled PM decisions will be taken, leading to overall factory efficiency.
Scheduling with PM is an important issue when talking about a fab wide fully

integrated systems. This idea is not thoroughly studied in the literature in semiconductor manufacturing. Cassady and Kutanoglu (2005) address this problem on a single machine. However, integrating of scheduling and APC systems requires a fab wide view of the problem.

After this overview on some interactions between scheduling and APC, let us address some new problems of integration of scheduling and APC. In Section 2.3, we discuss some of these possibilities.

2.3 Challenging integration problems

We recall that the literature on the integration of scheduling decisions and APC is not very explicit. There are some papers (Edgar et al. (2000), Holfeld et al. (2007), Purdy (2007)) on the integration of APC with the information system, in which the description stays at a structural level. Other disciplines are also concerned by the sociological difficulties that can hinder the idea of integration (Shobrys and White (2002)). We highlight in the following section, the importance of integrating scheduling decisions and APC information by identifying new promising problems.

2.3.1 Scheduling with APC (R2R/VM) constraints

A recipe corresponds to specifications on how a process should be executed on a tool (temperature, pressure, metal composition, etc). A Run-to-Run controller has to provide the right recipe conditions, for a given process, depending on the testing results obtained from feed forward and feedback metrology data. An essential calibration of the Run-to-Run controller is done when setting it up and, after that, the controller parameters will be regularly adjusted to compensate for the drifts and shifts in recipe parameters. If metrology results are not sent regularly, the R2R controller parameters become out of date. Consequently, the risk of discarding the processed lot/wafers is increased. Also, the needs of a virtual metrology (VM) lead to a similar constraint. A VM model is based on FDC (monitoring equipment status) results obtained from at least two measures, in order to provide approximate virtual metrology results.

The problem comes out from the fact that, with time, the Run-to-Run control loop parameters parameters have not to be updated for long time. In this case, the last recipe parameters will become non suitable for processing certain types of lots on the machine after a given interval of time, called *time* *threshold* in the sequel. Hence, lots should be sent regularly to the machine in order to keep the Run-to-Run control loop parameters updated by inspecting production lots/wafers and reporting the obtained results to the controller. More precisely, the problem does not lie in the idleness of a machine in a fab, but that the machine must process different types of products, thus the time between the arrivals of two lots (results of inspection tasks) of the same type is considered as an idle time for the Run-to-Run control loop parameters.

Therefore, a time constraint (*threshold*) could represent the R2R/VM requirements. This threshold is to be specified depending on several criteria such as the process type, the equipment, the maturity of control loop, etc. Scheduling with these constraints is a new challenging problem that is expected, to improve product quality, process and machine efficiency. This problem become more and more interesting if we consider the tremendous number of control loops that should be maintained in a fab as well as the various product types and equipment. Alternatively, it is recommended to keep all the control loops of a fab updated from a tool/process level point of view. However, it is complex and not always necessary to maintain all these control loops from a supervisory point of view.

Figure 2.1 describes the above addressed problem. Scheduling of jobs could become more effective in terms of product quality if the information and requirements of the APC system were taken into account. For example, information required by R2R control and Virtual Metrology leads to additional constraints on the scheduler and hence enhanced scheduling techniques should be developed. This problem will be addressed in the next chapters.

2.3.2 Scheduling with Equipment Health Factor

Equipment Health Factor (EHF) can be exploited to enhance scheduling. Let us recall that the EHF is an indicator associated to a tool to describe its state such as its level of reliability. On the other hand, a criticality indicator associated to a lot, wafer, product type (*family*) indicates the state of the considered element. Based on these indicators (equipment, lot), we can group tools/lots into categories as for example excellent, good, fair, etc.
More precisely, a *good* tool means that this tool has a good level of reliability, and a *good* lot means that this lot is not as critical as a lot of the *fair* category and so on.
The criterion that should be considered lies in the association of lots to tools, both possess different categories. Scheduling in this case become more challenging because if it is done without taking into account these indicators, a

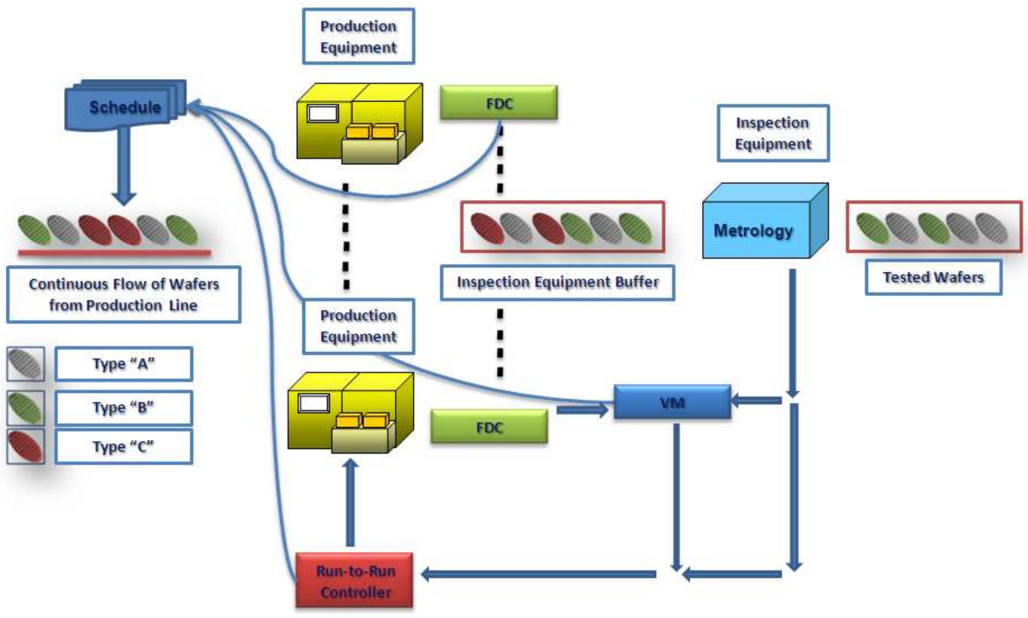


Figure 2.1: Scheduling with APC constraints.

yield loss is expected. The cost of assigning a lot to a tool is represented as a percentage of loss, probability of loss, yield, a combination of all of them, etc. The new challenging problem is how to schedule while minimizing such a cost i.e. maximizing the profit.
Figure 2.2 describes the mapping that can exist between lots and tool. While scheduling lots on the different machines, we may look to a mapping of lot/machine that minimizes certain risk. The later can be taken as a percentage of chip loss or/and a probability of loss. Whatever the criterion, the scheduling approach to propose should minimize a novel criterion. This problem will be treated in more details in Chapter 3.

2.3.3 Scheduling to maximize information in metrology

Metrology operations are usually considered as non-value added operations since they consume time and do not change the nature of the product. Available sampling policies are based on static or fixed criteria (Bousetta and Cross (2005)). The problem here lies in how to make such sampling not only dynamic, but also smart by benefiting from information on the routes of the lots. Briefly, knowing the sequence of machines on which a lot was processed,

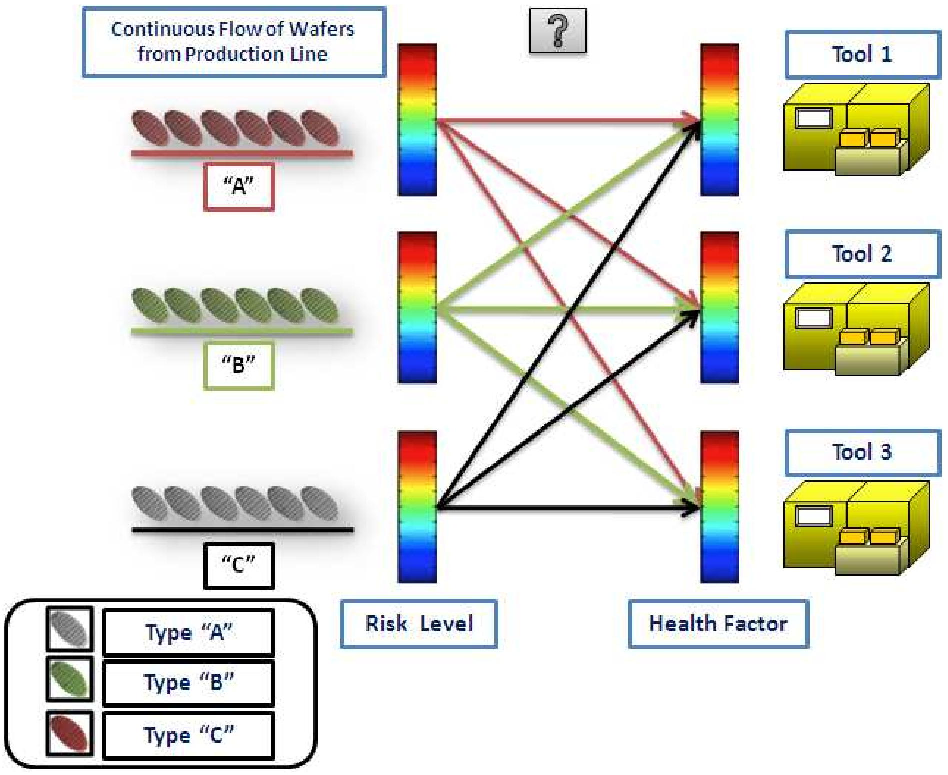


Figure 2.2: Scheduling with APC information (Equipment Health Factor).

we will then be able to decide whether this lot should be tested or not depending on the current risk levels of the associated machines.

Figure 2.3 explains how the route followed by a lot can be optimized to maximize the information on machines. By using the obtained information on which lots to select for measurement, advanced smart sampling techniques can be proposed (Dauzère-Pérès et al. (2010)).

2.3.4 Production and metrology scheduling

In general, the scheduling of production operations and the scheduling of metrology operations are done independently one from another. However, a strong link exists between production and metrology. For example, a lot of a given product type is processed on a production tool and then on a metrology tool. The processing of another lot of the same product type ideally should not be started before the end of the inspection of the previous lot. This is because the results obtained from metrology are used to compensate

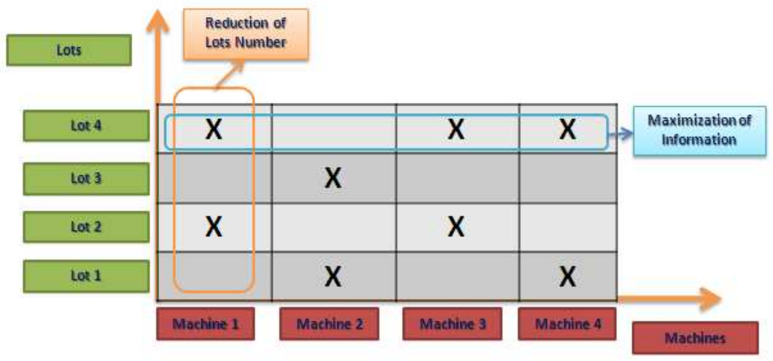


Figure 2.3: Scheduling while maximizing information on machines.

for process drifts and shifts (an APC need). As shown in Figure 2.4, in the first case (top), the processing of a lot of type A is started before the end of the related metrology operation, hence the wafers in the lot are at risk. However, if we shift the metrology operations so that the lot of type A comes before the inspection of a lot of type X, as in the same figure (bottom), then the wafers are no longer at risk.

Another possibility is to change the production schedule as shown in Figure 2.5 (bottom), the lot of type A starts after the end of the related metrology operation, hence wafers in the lot are not at risk. This change in production and/or metrology operation sequences illustrates the interaction between production and metrology. We believe that this problem is a good example of the integration of scheduling and APC systems. This problem was studied in Detienne et al..

2.4 Conclusion

In this chapter, we have proposed an overview of the benefits resulting from the integration of scheduling and APC systems. Possibilities of integration and new problems were discussed. From those, we will treat problems of scheduling with APC constraints and scheduling with EHF in the next chapters.
Integrating scheduling and APC information is a very challenging research problem, which combine different problem aspects, dimensions and research disciplines. Furthermore, active research collaborations between academia and industry also play a key role for the success of this topic.

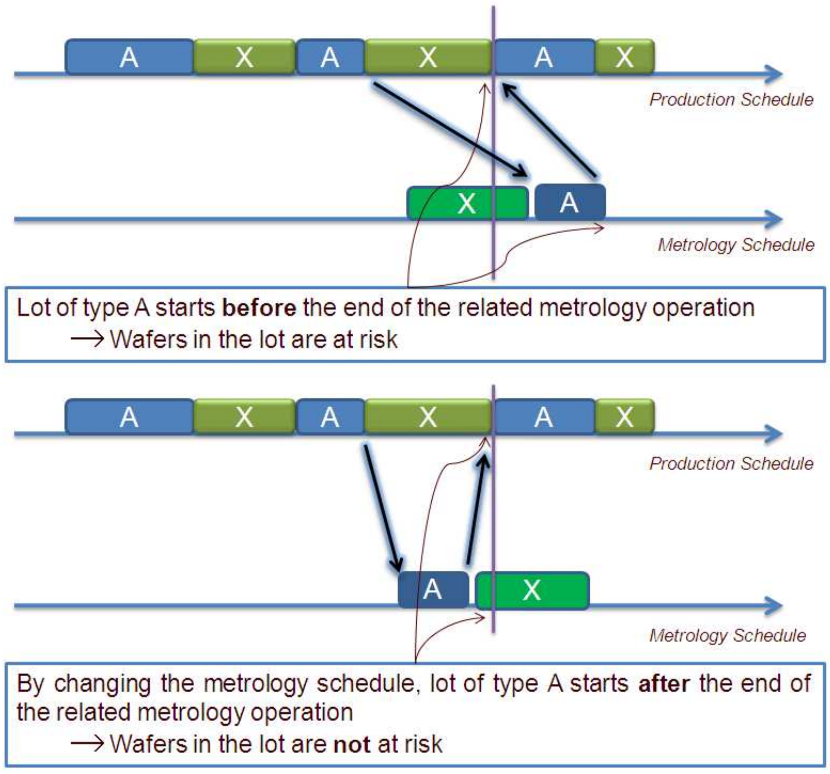

A
X
A
X
A
X
Production Schedule
X
A
Metrology Schedule
Lot of type A starts **before** the end of the related metrology operation
⟶ Wafers in the lot are at risk
A
X
A
X
A
X
Production Schedule
A
X
Metrology Schedule
By changing the metrology schedule, lot of type A starts **after** the end of the related metrology operation
⟶ Wafers in the lot are **not** at risk

Figure 2.4: Scheduling with interaction between production and metrology operations (1)

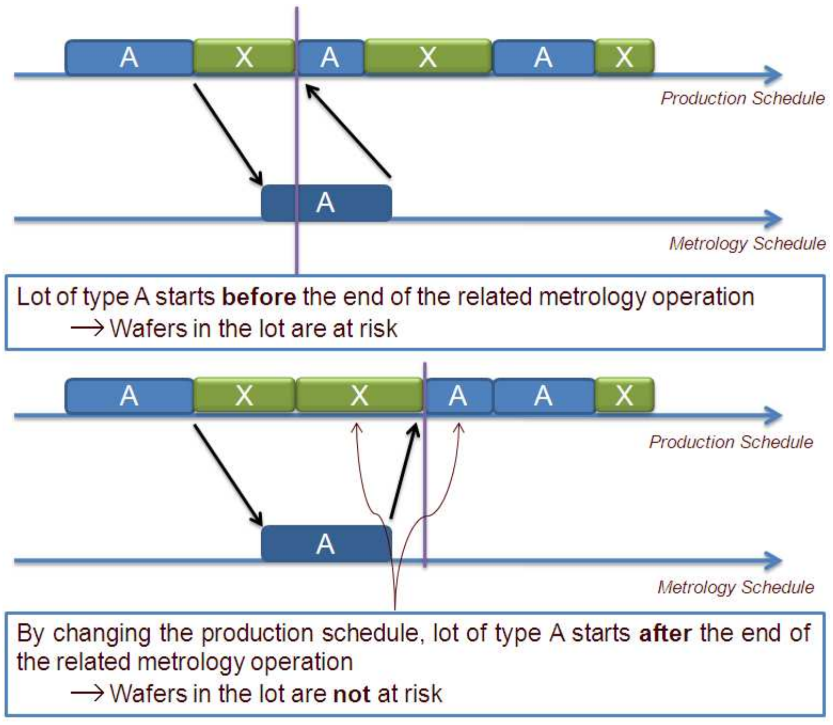

A
X
A
X
A
X
Production Schedule
A
Metrology Schedule
Lot of type A starts **before** the end of the related metrology operation
⟶ Wafers in the lot are at risk
A
X
X
A
A
X
Production Schedule
A
Metrology Schedule
By changing the production schedule, lot of type A starts **after** the end of the related metrology operation
⟶ Wafers in the lot are **not** at risk

Figure 2.5: Scheduling with interaction between production and metrology operations (2)

Chapter 3

Scheduling with Time Constraints and Equipment Health Factor

3.1 Introduction

Semiconductor manufacturing is getting more and more competitive and industries are looking for innovative strategies to improve productivity, decrease cost and enhance quality. Advanced Scheduling and Advanced Process Control (APC) systems support these objectives. Scheduling means assigning jobs to machines and sequencing jobs on machines to minimize some given objectives under a set of constraints. Hence, optimized scheduling helps to increase productivity. Process control is widely used to enhance the quality of products by compensating for process drifts and adjusting machine parameters. The collection of data at both machine and process levels helps in the detection of current process drifts and/or machine degradation, as well as in the prediction of possible faults. Scheduling and control could be considered as mutually related issues in semiconductor manufacturing. For example, to control, we may need information on scheduling, and to schedule in an effective way, we need information on which machines can process which operation.

Scheduling of lots has a direct impact on equipment utilization, cycle times, delivery times, etc. For example, effective scheduling decisions would send tasks to the right machines so as to avoid idle times and improve machine utilization. Moreover, semiconductor fabrication plants have characteristics that make scheduling a very complex issue (see Kumar (1993) or Moench et al. (2011) for instance). Given the re-entrant nature of manufacturing processes, scheduling is often locally optimized in each work area. We recall that Advanced Process Control (APC) aims at controlling processes and equipment to reduce variability, to increase equipment efficiency, or to collect and classify information on equipment to name a few. APC is usually associated to the combination of Statistical Process Control (SPC), Fault Detection and Classification (FDC), Run to Run control (R2R), and more recently Virtual Metrology (VM). Let us recall Figure 3.1 which shows the relation between scheduling and the information from the APC control system. Information on the needs of different APC control system components such as R2R and VM could be used to direct the scheduler towards better schedules in terms of quality.

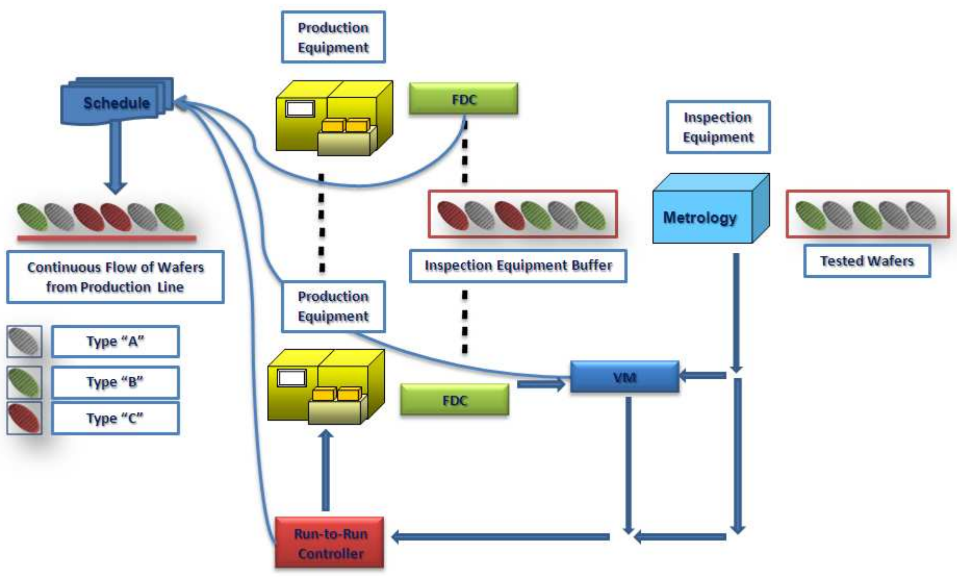


Figure 3.1: Scheduling with APC constraints (recall).

In semiconductor manufacturing facilities, a wafer is the chip holder at the end of the manufacturing process. Lots contain 25 wafers or less and are processed in various work areas with different characteristics. From now on, lots will be called **jobs**, and lots of the same product type will be called **job family**. While investigating some works that are found in the literature, we find that Kubiak et al. (1996) study the problem of scheduling a reentrant job shop with different job families. They show that the shortest processing time (SPT) job order is optimal for the single machine reentrant shop under certain assumptions. An example can be found in the photolithography area that can be seen as a scheduling problem on parallel machines with job family setups (also called s-batching). A setup is required before starting the first job of a family, but no setup is necessary between two jobs of the same family. For example, the change of reticles in the photolithography workshop necessitates a family dependent setup time. Although research has been performed on this problem, very little has been done to integrate APC constraints.

In R2R control for instance, a R2R controller uses data from past process runs to adjust settings for the next run as presented in Musacchio et al. (1997). Note that a R2R controller is associated to one machine and one job family. A machine can usually process a limited number of job families, that are said to be *qualified* on the machine. Machines are thus non-identical in terms of qualifications. Hence, a machine is qualified to process a job family if and only if it is *eligible* (capable) and configured to process such a job family. Machine eligibility, and availability will be addressed later in this chapter. Further, in order to keep its parameters updated and valid, a R2R control loop should regularly get data. This imposes an additional constraint on scheduling, since jobs of the same family have to be scheduled within a maximum time interval on each machine on which the family is qualified. The value of this time interval (*threshold*) depends on several criteria such as the process type (critical or not), the equipment type, the stability of the control loop, etc. If this time threshold/constraint is not satisfied, a qualification run is required to be able to process again the job family on the machine. This leads us to define a new scheduling problem that integrates the previously described time threshold as a hard constraint, when we assume that a qualification run cannot be performed within the scheduling horizon. We first consider the problem of scheduling lots (jobs) of different product types on parallel machines, where not all machines are able to process all job families (non-identical machines). A special time constraint, associated to each job family, should be satisfied for a machine to remain qualified for processing a job family. This constraint imposes that the time between the execution of two consecutive jobs from the same family on a qualified machine must not exceed the time threshold of the family. Otherwise, the machine becomes disqualified.

Figure 3.2 illustrates this time constraint. In the first two cases, a job of a given family is started during the time interval corresponding to its family, and hence the machine will still be qualified to process jobs of the same family for another time interval. The third case represents the situation where a machine is no longer qualified to process such a job family, and this is because no job is scheduled during the considered interval. This problem comes from semiconductor manufacturing, when Advanced Process Control (APC) constraints are considered in scheduling problems, for example in the photolithography area as mentioned earlier.

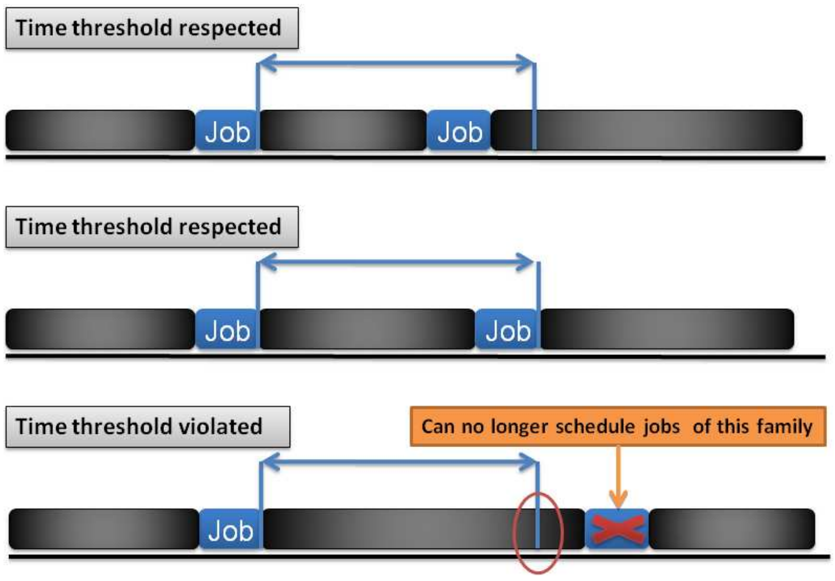


Figure 3.2: Time constraint.

Moreover, considering machine qualification in *PTC* leads us to introduce a new concept related to the assignment of job families to qualified or semi-qualified machines. In our problem, called *PTC* for Problem with Time Constraint, machine qualification is considered as a boolean parameter *i.e.* a machine is either qualified (possibility of processing with a perfect outcome) or disqualified (impossibility of processing). However, to be more realistic, this notion may be seen as a level of qualification. This can be modeled by a certain risk associated to the assignment of a job family to a machine. This risk could be a function of the Equipment Health Factor (EHF) and the Wafer Quality Indicator (WQI) previously described in Chapter 2. We define *PEHF* (Problem with Equipment Health Factor) as an extension of *PTC*, where the global risk associated to assigning jobs to machines is also minimized. In *PEHF*, we assume that a machine is disqualified for a job family if the risk associated to assigning this machine to this job family is larger than a given upper limit.
This chapter is organized as follows. Section 3.2 provides a literature review on related problems. Section 3.3 gives an overview on multicriteria optimization since our problems are multicriteria. Both problems are described,

defined and modeled in Section 3.4 (PTC), and Section 3.5 ($PEHF$), and will be analyzed and solved in Chapter 4.

3.2 Literature review

There are very few articles which deal with scheduling decisions while integrating APC constraints. The impact of APC on scheduling performances is analyzed by Li and Qiao (2008). They also study the scheduling of job families on parallel machines. However, they consider that machines are identical, that qualification runs can be scheduled and that the threshold between two jobs of the same family is given in number of jobs. We consider non-identical parallel machines and assume that qualification runs cannot be scheduled and will be performed after the scheduling horizon. The problem becomes more complicated, since the assignment of jobs to machines is critical to avoid qualification runs. Finally, we consider a threshold expressed in time instead of number of jobs. Both threshold types are actually relevant and are related. Cai et al. (2011) study the interaction between scheduling and APC on one machine, with setup times between two job families, and a qualification run when the R2R constraint is not respected. They show that a single machine makespan problem with multiple job families is NP-hard. Another example of the integration of APC constraints in scheduling decisions can be found in Detienne et al. (2012), where measurement operations are optimally scheduled to minimize the risk of losing products in jobs.
This chapter addresses two new scheduling problems (PTC and $PEHF$). In PTC, there is a time constraint on jobs of the same family i.e. the time interval between two consecutive jobs of the same family should be smaller than a given threshold. The goal is to schedule job in families on parallel machines with information on APC system while minimizing the number of loss in machine qualifications. As mentioned above, this constraint is inspired from the needs of APC systems, and in particular Run-To-Run (R2R) control loops for a given product type on a machine, that require to regularly collect data for product types on machines. $PEHF$ is an extension of PTC in which the states of machines is taken into consideration. In what follows, we review some existing topics related to our problems.

3.2.1 Dynamic dead-lines/due dates

A deadline d is a point in time by which a job must absolutely complete. Scheduling with dynamic deadlines exists in the literature under various topics and rarely in the domain of semiconductor manufacturing. Studies on dynamic deadlines can be found in mobile communications.

Somasundara et al. (2007) study the problem of Mobile Element Scheduling (MES). The mobile elements visit the nodes of a wireless sensor network to collect their data before their buffers are full. In addition, as soon as a node is visited, its deadline (time before which it should be revisited to avoid buffer overflow) is updated. Thus, deadlines are *dynamically* updated as the mobile element performs the job of data gathering. The idea is to find a schedule for a controlled mobile element so that there is no data loss due to buffer overflow. Other examples of dynamic deadlines problems in the same domain include mobile element for data collection, battery charging, and calibration (Kallapur and Chiplunkar (2010)).

Further, Caccamo et al. (1999) address the problem of scheduling hybrid tasks in a shared resource environment (hard periodic and soft aperiodic) with dynamic deadlines. They develop an algorithm which finds the optimal solution for a schedule of hybrid tasks on shared resources. The problem is basically a problem of task scheduling in computer operating systems. The problem of lot release control and scheduling in wafer fabs producing multiple products with due dates, was tackled in Kim et al. (1998). The authors suggest several new rules to minimize mean tardiness. They show that new dispatching rules work better in terms of tardiness of orders compared to existing rules such as EDD (Earliest Due Date) for multi-machine scheduling. By analogy, the idea of dynamic deadlines exists in our problem (PTC) under the form of job family thresholds, where each family threshold creates a deadline at the machine qualification level. This threshold is dynamically updated once a job of a given family is scheduled on a qualified machine.

3.2.2 Scheduling on parallel machines

Parallel machine scheduling problems are frequent in semiconductor manufacturing. A wafer fab can be modeled as a complex job shop (Mason et al. (2002)), which contains unrelated parallel machines with sequence-dependent setup times and dedications, parallel batch machines, re-entrant flows, and ready times for the jobs (Moench et al. (2011)). Classical scheduling objective functions include (Pinedo (2009)) the minimization of makespan (C_{max}), total weighted tardiness ($\sum w_jT_j$), etc. Another classical scheduling objective function is the minimization of the total completion time ($\sum C_j$). Rules such as SPT (Shortest Processing Time) are used to solve the problem. When the total weighted completion time $\sum w_jC_j$ (also $\sum C_j$ on non-identical machines) is to be minimized on parallel machines.

Unrelated parallel machines

Among parallel machine scheduling problems, one of the general cases is when the machines are *unrelated*. In this case, the processing time of each job depends on the machine to which it is assigned. Ruiz and Andrés (2007) study the problem of scheduling unrelated parallel machines with resource-assignable sequence dependent setup times. They address a problem where the duration of the setups depend on assignable unrelated machines. They develop some fast heuristics to tackle this problem and a mathematical model is given and the sum of completion times of jobs is minimized.

Meta-heuristics were also used to tackle the problem of scheduling unrelated parallel machines. For example, Kim et al. (2002) address the problem of scheduling unrelated parallel machines with sequence-dependent setup times, using Simulated Annealing (SA) to minimize total tardiness. The developed SA with different types of neighborhoods shows significant performance.
Moreover, the problem of scheduling unrelated parallel machines with classical objective functions, such as minimizing the makespan or the total weighted tardiness, was also studied in the literature (Anagnostopoulos and Rabadi (2002); Lin et al. (2009); Zhou et al. (2007); Na et al. (2006)). Different heuristics and meta-heuristics (Simulated Annealing, Ant Colony Optimization,etc) were proposed to tackle this problem.

Some basic algorithms

List scheduling *List Scheduling* (LS) has been well known since more than half a century ago. In this algorithm, jobs are fed from a pre-specified list and, whenever a machine becomes idle, the first available job on the list is scheduled and removed from the list, where the availability of a job means that the job has been released and, if there are precedence constraints, all its predecessors have already been processed.
Because of its simplicity and the fact that any optimal schedule can be constructed by LS with an appropriately chosen list, LS is by far the most popular scheduling approach (Anderson (2004)).

LPT and SPT If the jobs in LS are sorted in order of non increasing processing times, then the resulting algorithm is known as *Largest Processing Time* (LPT). On the other hand, if the jobs in LS are sorted in order of non decreasing processing times, then the resulting algorithm is known as *Shortest Processing Time* (SPT). Contrary to LPT, SPT tends to better minimize the total completion time $\sum C_j$.

3.2.3 Set-up times

A setup is a non-productive period of time which usually models operations to be carried out on machines after processing a job to leave them ready for processing the next job in the sequence. An extended survey on scheduling problems with setup times or costs is done in Allahverdi et al. (2008). The authors provide an extensive review of the scheduling literature on models with setups covering more than 300 papers. They classify problems with batching and non-batching considerations, and with sequence-independent and sequence-dependent setup times. They also categorize the literature according to shop environments, including single machine, parallel machines, flow shop, no-wait flow shop, flexible flow shop, job shop, open shop, and others.

In addition, scheduling jobs on parallel machines with sequence-dependent family setup times is also studied in Eom et al. (2002). The authors propose a three-phase heuristic to minimize the total weighted tardiness of a set of tasks with known processing times, due dates, weights and family types for parallel machines. They consider the case of identical machines in a Liquid Crystal Display (LCD) manufacturing process where the setup time is longer than the processing time. Schaller et al. (2000) study the problem of scheduling a flowline manufacturing cell with sequence dependent family setup times. The objective is to minimize the makespan. The authors show that the problem is NP-hard in the strong sense, and they develop several heuristics. In this chapter, we consider non-identical parallel machines and minimize a bi-criteria objective function where the sum of completion times is one criterion.
Setup times can be classified in two categories: Batch and non-batch setup times.

Batch setup times

A batch is a set of jobs of the same family. While families of jobs are supposed to be given in advance, batch formation is a part of the decision making process. A batch setup time occurs when jobs are processed in batches, and a setup time precedes the processing of each batch. The batch setup time can be machine dependent and/or sequence dependent. It is sequence dependent if its duration depends on the families of both the current and the immediately preceding batches, and is sequence independent if its duration solely depends on the family of the current batch to be processed.

Non-batch setup times

In a non-batch processing environment, a setup time is incurred prior to the processing of each job. It can be seen as a batch setup time in which each family consists of a single job. Figure 3.3 describes this classification.

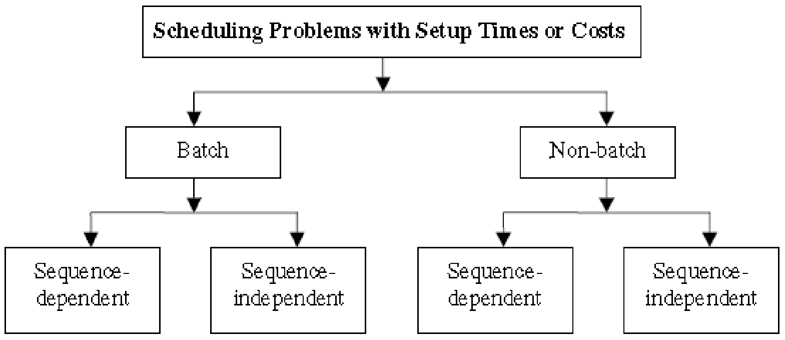


Figure 3.3: Classification of scheduling problems with setup times.

3.2.4 Machine eligibility and availability

A machine might become unavailable due to machine breakdowns or various other reasons and in particular preventive maintenance. However, a machine might have different capability to process different jobs. In our research, we are interested in machine eligibility, where a qualified machine is eligible to process a given job family, but may lose its qualification (PTC) if a given time constraint is violated. The notion of "semi-qualification" of a machine is used in $PEHF$ where assigning a job of a given job family to a machine is no longer a matter of a total qualification or not, but rather a degree of confidence in the machine reliability represented modeled as a risk of assignment.

Parallel machines are common in manufacturing industries. Parallel machine scheduling problems with machine eligibility restrictions are of considerable interest for industry in general and in particular in semiconductor manufacturing (Huang and Yu (2010)). They generalize scheduling problems on a

single machine, on identical parallel machines and on uniform parallel machines (Guo and Liu (2010)). For example, in semiconductor manufacturing, the wafer testing area is usually divided into five workcenters: Voltage and current test, wafer test or also known as probe, back-grind, visual inspection and ship (Centeno and Armacost (1997)). A wafer testing work-center involves a number of testing machines in parallel with different capabilities.
The problem of scheduling uniform parallel machines with machine eligibility restrictions to minimize total weighted tardiness (TWT) is studied in Guo and Liu (2010). The authors presented a new heuristic based on local search on each machine. The experimental results have shown that the proposed heuristic is very effective method for this problem. In addition, Huang and Yu (2010) develop a novel heuristic to minimize makespan for parallel machines with eligibility constraints while considering machine flexibility. Their results show that their heuristic can achieve exact or nearly exact solutions. Moreover, Liao and Sheen (2008) address the same problem but with machine eligibility and availability constraints i.e. the machines are not continuously available (availability) at all times and each job can only be processed on specified machines (eligibility).
Further, the problem of minimizing maximum lateness in a parallel machine environment with release dates and machine eligibility restrictions is addressed in Centeno and Armacost (1997). The authors develop a scheduling algorithm for a wafer test work-center taking into consideration real manufacturing conditions and constraints. The results on real data showed a significant performance improvement over the previously used schedule. Also Sheen et al. (2008) study the problem of scheduling independent jobs on identical parallel machines with machine availability and eligibility constraints to minimize the maximum lateness.

3.2.5 Scheduling techniques

We recall that every NP-complete problem can be solved by exhaustive search. Known NP-complete problems include scheduling under precedence constraints. While trying to solve these problems, many techniques were developed. Essentially, most approaches for semiconductor manufacturing scheduling problems can be classified into four categories: Heuristic rules, mathematical programming techniques, neighborhood search methods, and artificial intelligence techniques (Gupta and Sivakumar (2004)).
For our problems, we used mathematical programming, heuristic rules and a neighborhood search method.

3.3 Multicriteria Optimization

Most problems in nature have several (possibly conflicting) objectives to be satisfied. Many of these problems are frequently treated as single-objective optimization problems by transforming all but one objective into constraints. Most scheduling problems in the literature consider only one optimization criterion. However, many real industrial problems require that two or more criteria are simultaneously optimized. These objectives are usually conflicting. Therefore, there is no optimal solution that optimizes all the criteria simultaneously. For example, minimizing the sum of completion times for all jobs is one criterion that is antagonistic with minimizing the sum of losses in machine qualifications. The objective function that we study in PTC and $PEHF$ is then of a multicriteria nature. The Multi Objective Optimization Problem (MOOP) (also called multicriteria optimization, multiperformance or vector optimization problem) can be defined as the problem of finding:
A vector of decision variables which satisfies constraints and optimizes a vector function whose elements represent the objective functions. These functions form a mathematical description of performance criteria which are usually in conflict with each other. Hence, the term "optimize" means finding such a solution which would give the values of all the objective functions acceptable to the decision maker (Eschenauer et al. (1990), Coello (2011)).

3.3.1 Linear and Nonlinear Multi-Objective Optimization Problems

If all objective functions and constraint functions are linear, the resulting MOOP is called a multi-objective linear program (MOLP). PTC and $PEHF$ are MOLPs. Like linear programming problems, MOLPs also have theoretical properties. However, if any of the objective or constraint functions are nonlinear, the resulting problem is called a nonlinear multi-objective problem. Unfortunately, for nonlinear problems, solution techniques often do not have convergence proofs Deb (2001). There exist different methodologies that were developed to deal with multicriteria optimization. Pareto optimality is a fundamental notion in multicriteria optimization.

3.3.2 Definition of optimality

Let $S \subset \mathbb{R}^{\mathbb{Q}}$ be the set of solutions and $\tilde{Z} \subset \mathbb{R}^{\mathbb{K}}$ the image in the criteria space of S by K criteria Z_j. The order structure associated with $\mathbb{R}^{\mathbb{K}}$ is, $\forall x, y \in \mathbb{R}^{\mathbb{K}}$:

$$x \leq y \Leftrightarrow x_i \leq y_i, \forall i = 1, \ldots, K \tag{3.1}$$

$$x = y \Leftrightarrow x_i = y_i, \forall i = 1, \ldots, K \tag{3.2}$$

This order defines a partial preorder, valid for $K \geq 2$. Note that, for single criterion optimization problems ($K = 1$), the structure associated with R is a total preorder, i.e. there is no incomparability between two solutions. Thus, in the single criterion case, the definition of an optimal solution is straightforward. In the multicriteria case, this definition is no longer trivial because a solution minimizing simultaneously all criteria rarely occurs. We then use a more general definition of optimality: Pareto optimality (T'Kindt and Billaut (2006)).

Definition - weak Pareto optimality. *$x \in S$ is a weak Pareto optimum, also called a weakly efficient solution, if and only if $\nexists y \in S$ such that $\forall i = 1, \ldots, K; Z_i(y) < Z_i(x)$. We note WE the set of weak Pareto optima of S. The set WE defines in the criteria space the trade-off curve, also called the efficiency curve.*

Definition - strict Pareto Optimality. *$x \in S$ is a strict Pareto optimum, also called an efficient solution or a strict efficient solution, if and only if $\nexists y \in S$ such that $\forall i = 1, \ldots, K; Z_i(y) \leq Z_i(x)$ with at least one strict inequality. We note E the set of strict Pareto optima of S and we have $E \subseteq WE$.*

3.3.3 Determination of Pareto optima

Pareto optima correspond to "best trade-off" solutions between different conflicting criteria. Only the decision maker can choose the most satisfactory solution for his problem, among the set E (or WE). Three occasions where the decision maker can intervene: before, during or after the resolution process. A general category of methods can be associated to each of these occasions:

- The methods enabling the decision maker to intervene before the resolution process are called a priori. In the a priori methods, the resolution process cannot be performed without the decision maker having provided a set of information, as for example the value of the weights of the criteria for the minimization of a linear combination of criteria. Determining these parameters constitutes a problem itself, which requires the use of a decision aid method.

- The methods enabling the decision maker to intervene during the course of the resolution process are called interactive. In the interactive methods, the resolution process is iterative. Each iteration provides the decision maker a solution, which is not necessarily a Pareto optimum. He then orients the process by providing, directly or indirectly, new values for the parameters of the problem.

- The methods enabling the decision maker to intervene after the resolution process are called a *posteriori*. A posteriori method aims at providing the decision maker with an exhaustive set of Pareto optima, among which belongs the most satisfactory solution. The set of Pareto optima suggested to the decision maker depends on the properties of the problem.

There are different families of methods to find Pareto optima. The list below provides a summary of some of these methods. The choice of a method necessitates a trade-off between the quality of the calculable solutions and the ease of its application. Let us recall the notation $\alpha|\beta|\gamma$ introduced to classify scheduling problems by Graham et al. (1979). The γ field contains the criteria to be optimized. In the case of multi-objective scheduling, a precise description of the field γ is also given by (T'Kindt and Billaut (2006)). This field helps to differentiate between existing methods to find Pareto optima. These methods include:

- **Determination by linear combination of criteria.** In this approach, we aggregate the values of different criteria to finally obtain one single indicator. The coefficients of each criterion helps in determining the direction of solution searching. In this case, the γ field is defined as: $F_l(Z_1, \ldots, Z_K)$. The modeling of PTC and $PEHF$ includes an objective function which is a linear combination of the corresponding criteria. However, by setting the weight of one of the criteria in such a way that it dominates completely the other criteria, then the criteria are considered in lexicographical order.

- **Determination by lexicographical approach.** In a lexicographical approach, the criteria are optimized one after the other. This means that once the first criterion is optimized, the second is then optimized while inhibiting the first to be degraded and so on. In this case, the γ field is defined as: $Lex(Z_1, \ldots, Z_K)$ (Lexicographic).

- **Determination by means of the ϵ-constraint approach.** Only one criterion Z_u is optimized in this approach, where all other criteria are taken as constraints bounded by a certain value. In this approach, the γ field is defined as: $\epsilon(Z_u|Z_1, \ldots, Z_{u-1}, Z_{u+1}, \ldots, Z_K)$.

- **Determination by the goal-programming approach.** This approach aims at finding a solution that satisfies the objectives for each criterion. In this case, the γ field is defined as: $GP(Z_1, \ldots, Z_K)$.

More methods and details are provided in T'Kindt and Billaut (2006), where the authors cover the theory, models and algorithms of multicriteria scheduling. In this chapter, we focus on the determination of Pareto optima by linear combination of criteria. We believe that these coefficients are more realistic for industrial purposes since the decision maker can choose different weights for each criterion of the objective function. Consequently, the ease of implementation this method is also a strong motivation.

3.4 Scheduling Job Families on Non-Identical Parallel Machines with Time Constraints (*PTC*)

3.4.1 Definition of PTC

Let us recall that, in *PTC* lots (jobs) of different product types (job family) are scheduled on parallel machines, where not all machines are able to process all job families (non-identical machines). We consider a time constraint on jobs of the same family, i.e. the time interval between two consecutive jobs of the same family should be smaller than a given time threshold. As mentioned in the previous section, this constraint is inspired from the needs of APC systems, and in particular Run-To-Run (R2R) control loops for a given product type on a machine, that require to regularly collect data for product types on machines. We assume in our problem that this qualification run cannot be performed within the scheduling horizon. The problem becomes more complicated, since the assignment of jobs to machines is critical to avoid qualification runs.

Our objective is to schedule, on a horizon discretized in T periods, a set N of jobs of different families on a set M of parallel machines. The set of job families is denoted F, and $f(i)$ is the family of job i. We assume that the processing times p_f of all jobs in family f are equal. Machines are not qualified to process all jobs families. The qualification of a machine may be lost at a certain point in time due to a change in the level of confidence on the machine. A setup time $s_{f'}$ on a machine is necessary to change from one job of a family f to another job of family f', where $f \neq f'$. Finally, Run-To-Run control constraints are considered through a parameter γ_f, which corresponds to the maximum time interval (called time threshold in the sequel) between the processing of two jobs of family f on a qualified machine. Usually, if this constraint is not satisfied, a qualification run will be required to qualify again the machine for f. In the sequel, we consider that the machine will not be available to process any job of family f if the qualification cannot be maintained. The objective is to optimize a scheduling criterion, the sum of the completion times, while minimizing the number of disqualifications of families on machines, i.e. a bicriteria scheduling problem. According to the $\alpha|\beta|\gamma$ notation introduced to classify scheduling problems by Graham et al. (1979), this problem is noted $P_m|ST_{si,b}|\sum C_j$.

3.4.2 Notations

Parameters:

The parameters are:

T: Number of periods in the time horizon,
N: Set of jobs,
M: Set of machines,
F: Set of job families,
$M(f)$: Set of qualified machines to process jobs in family f ($M(f) \subset M$),
p_f: Processing time of jobs in family f,
s_f: Setup time of jobs in family f,
γ_f: Time threshold for job family f,
$f(i)$: Job family of job i ($f(i) \in F$).

Decision variables

The decisions variables are:

$x_{i,t}^m = 1$ if job i starts at period t on machine m, and 0 otherwise,
C_i: Completion time of job i,
$y_{f,t}^m = 1$ if the time threshold is not satisfied for family f on machine m at period t, i.e. a qualification run is required, and 0 otherwise.

It is important to recall that, if the time threshold γ_f is not satisfied for job family f on machine m, the qualification run required on machine m cannot be performed within the time horizon. In this case, we suppose that no job of family f can be processed on m. Moreover, these notations are still valid when modeling $PEHF$. Additional notations will later be defined.

3.4.3 Complexity

Let us recall that a machine is said to be *qualified* to process a given job family if it satisfies the necessary conditions to process this job. Our problem consists of the scheduling of $|F|$ job families, where $|F|$ is the cardinal of the set of job families F, on M non-identical parallel machines (each machine has its own qualifications), with s_f as the setup time of family $f \in F$ and p_f as the associated processing time. A *time interval* is associated to each family during which at least one job of this family must be scheduled. We call this time constraint *threshold*. The value of the family threshold is given by γ_f, and n_f is the number of jobs in family f. Initially, we must send a job of family f to a qualified machine m during the interval $[0, \gamma_f]$. Otherwise, the machine will not be available anymore to process such a job family (the machine is disqualified). The objective function of our problem is a bi-criteria one, in which we aim to minimize both the sum of completion times of families ($\sum C_f$) and the number of machine disqualifications represented by ($\sum Y_f$).

Case of parallel machines

Let us consider the following particular case of our problem where the setup time is set to zero for all job families ($s_f = 0, \forall f \in F$). The threshold associated to each job family defined by γ_f is considered as the deadline d_f of a given family. Let us recall that all jobs of a family have the same deadline (threshold). The deadlines (thresholds) of all families are considered to be equal to the time horizon, i.e. $d_f = \gamma_f = T, \forall f \in F$. We assume that all machines are qualified to process all job families. Hence, the qualification part in the objective function has no effect, and the objective function becomes the classical known function of minimizing the sum of completion times. Moreover, the machines are identical in terms of qualifications. Therefore, the problem is reduced to scheduling n jobs on m machines, with p_f as the processing time of job family f, and ($\sum C_f$) as the objective function. Webster (1997) proved that this problem is unary NP-hard, and therefore, our problem is NP-hard.

Case of a single machine

Another special case that could be deducted is the case of a single machine with time lags. Kum (1992) performs a thorough analysis of scheduling problems on one machine with generalized precedence constraints. Wikum et al. (1994) show that the problem of single machine scheduling with maximum delays, precedence constraint and minimizing the sum of completion times, is NP-hard in the strong sense. The proof is based on a reduction of a problem studied in Kum (1992). A summary of the complexity of scheduling problems with delays is given in Table 3.1.

Table 3.1: Complexity classification of min/max, precedence constrained scheduling problem with completion time objective function

Type of Delays	Objective Function	Complexity
min delays	$\sum C_j$	$O(k^3 lgk)$
min delays	$\sum w_j C_j$	NP-Hard
max delays	$\sum C_j$	NP-Hard in the strong sense

By taking the completion time as an objective function while considering that all machines are qualified, setting family thresholds as the maximum time lag per family (jobs are grouped in families), and considering only the case of a single machine, we find that the problem discussed above is a particular case of our problem.
Below, we consider an instance of PTC and show that it is equivalent to a general instance of the problem described in Wikum et al. (1994).

NP-hardness We prove the NP-completeness of PTC by polynomial transformation from the problem of Maximum Delay Precedence constraints with total completion time - MDP (1|max delays, k 1-chains|$\sum C_j$ which is proved to be NP-hard in [Kum (1992),Wikum et al. (1994)]). In what follows, we provide an instance of our problem:

> INSTANCE: A set of jobs $N = \{1,\ldots,n\}$, grouped in $f \geq 2$ families and F is the set of families such that $F = \{1,\ldots,f\}$, processing requirement $p_k \in Z_0^+ \ \forall\ k \in F$ (where $\forall i, j \in N$, $f(i) = f(j)$ implies $p(i) = p(j)$ i.e., jobs of the same family have the same processing time), family thresholds γ_k for $k \in \{1,\ldots,f\}$,

setup times of families s_k for $k \in \{1,\ldots,f\}$. One single qualified machine m which is qualified to process all family types, and a positive integer B.

QUESTION: Is there a one-machine schedule for N that places all the jobs such that any two consecutive jobs of the same family are not separated by more than γ_f , and such that the sum of completion times, is equal or less than B ($\sum_{j \in N} C_j \leq B$)?

PTC is in NP since one can check in polynomial time if a schedule has a sum of completion times less than or equal to B and satisfying that two consecutive jobs of a same family are separated by at most γ_f. The NP-completeness of *PTC* is proved by polynomial transformation from the *MDP* problem (Kum (1992)). Below a general instance of *MDP*:

INSTANCE: Job set $J = \{J_1, \ldots, J_k\} \bigcup \{*\}$, processing requirement $p_j \in Z_0^+ \ \forall J_j \in J$, precedence relation P on J of the form $P = \{\langle J_j, * \rangle,\ j = \{1,\ldots,k\}\}$, maximum delays u_j for $j = 1, \ldots, k$, where each delay is either infinite or a non negative integer, and a positive integer Y.

QUESTION: Is there a one-machine schedule for J (i.e., a function $\sigma : J \rightarrow Z_0^+$) that satisfies the maximum delay precedence constraints (i.e., $\sigma(*) - C_j(\sigma) \leq u_j$, where $\sigma(*)$ is the start date of the last job, $C_j(\sigma) = \sigma(j) + p_j \ \forall j = 1, \ldots, k$) and such that the sum, taken over all $J_j \in J$, of $C_j(\sigma)$ is Y or less?

Theorem 1. *Problem PTC is NP-complete.*

Proof. An instance of *PTC* leads to a general instance of *MDP* as follows:

- $J = F$, i.e. the set of jobs is the set of families when there is only one job per family,
- $p_j = p_f$, $\forall j \in J$,
- $s_f = 0$, $\forall f \in F$,
- $u_j = \gamma_f$, $\forall j \in J$,
- $Y = B$.

Suppose there is a “Yes” answer for PTC. Without loss of generality, we impose precedence constraints on jobs in each family. This will not change the characteristic of the problem since all jobs of the same family have the same processing time and jobs which belong to the same family are interchangeable. In addition, whenever the time threshold γ_f is respected, it ensures that the maximum delay precedence constraints (i.e., $\sigma(*) - C_j(\sigma) \leq u_j$, where $C_j(\sigma) = \sigma(j) + p_j \; \forall j = 1, \ldots, k$) is satisfied ($u_j = \gamma_f$), and therefore a “Yes” answer to MDP. $\square$

3.4.4 Mathematical programming models

In this section, mathematical programming models to tackle PTC are introduced. A job based mathematical model and a family based mathematical model are presented. The evolution of the family based mathematical model is discussed and the final model is given. The objective function for PTC is to minimize a scheduling criterion (sum of completion times) and a control criterion (sum of losses in machine qualifications).

Job-based model ($IP1$)

A first model ($IP1$) can be written as follows, where each job i is considered separately:

$$\sum_{m \in M(f(i))} \sum_{t=1}^{T-p_{f(i)}+1} x_{i,t}^m = 1 \quad \forall i \in N \tag{3.3}$$

$$\sum_{m \in M(f(i))} \sum_{t=1}^{T-p_{f(i)}+1} t.x_{i,t}^m + p_{f(i)} \leq C_i \quad \forall i \in N \tag{3.4}$$

$$\sum_{i \in N; f(i)=f} \sum_{\tau=t-p_{f(i)}+1}^{t} x_{i,\tau}^m \leq 1 \quad \forall t = 1 \ldots T, \forall f \in F, \forall m \in M(f) \tag{3.5}$$

$$\sum_{\tau=t-p_{f(i)}-s_{f(j)}+1}^{t} x_{i,\tau}^m + \sum_{\tau=t-p_{f(j)}-s_{f(i)}+1}^{t} x_{j,\tau}^m \leq 1 \quad \forall t = 1 \ldots T, \forall (i,j) \in N \times N \text{ s.t. } f(i) \neq f(j), \forall m \in M(f(i)) \cap M(f(j)) \tag{3.6}$$

$$\sum_{i \in N; f(i)=f} \sum_{\tau=t-\gamma_f+1}^{t} x_{i,\tau}^m + y_{f,t}^m = 1 \quad \forall f \in F, \forall m \in M(f), \forall t = \gamma_f \ldots T \tag{3.7}$$

$$y_{f,t-1}^m \leq y_{f,t}^m \quad \forall t = 2 \ldots T, \forall f \in F, \forall m \in M(f) \tag{3.8}$$

$$x_{i,t}^m \in \{0,1\} \quad \forall t = 1 \ldots T, \forall i \in N, \forall m \in M(f(i)) \tag{3.9}$$

$$y_{f,t}^m \in \{0,1\} \quad \forall f \in F, m \in M(f) \tag{3.10}$$

Constraint (3.3) guarantees that each job is scheduled once and only once on a machine in the scheduling horizon. Constraint (3.4) is used to determine the completion time of each job. Constraints (3.5) and (3.6) model the fact that only one job is processed at a time on a machine. Constraint (3.5) is written for jobs of the same family, i.e. for which no setup time is required, whereas Constraint (3.6) is associated to pairs of jobs of two different families for which setup times are necessary. Constraint (3.7) ensures that either the time threshold is always satisfied for a job family f qualified on machine m, or a qualification run is necessary, i.e. $y^m_{f,t} = 1$. Constraint (3.8) guarantees that, if a machine is disqualified at period t, then it is also disqualified in the following periods. Constraints (3.9) and (3.10) force variables $x^m_{i,t}$ and $y^m_{f,t}$ to be binary.

When analyzing the problem, it is possible to see that, since there are no release dates on jobs and jobs of the same family have the same processing time, all jobs in a family can be interchanged in an optimal solution. Let us denote by $x^m_{f,t}$ the decision variable that takes value 1 if a job of family f starts at period t on machine m and 0 otherwise, n_f the number of jobs in family f, and C_f the sum of the completion times of jobs in f. Model $IP2$ below is equivalent to $IP1$. Numerical tests were conducted to show that the family based model $IP2$ dominates the job based model $IP1$ in terms of execution times. In Chapter 4, numerical experiments conducted on both models are presented. The results in Chapter 4 shows that, for a set of generated instances, $IP2$ solves on average the instances 10 times faster than $IP1$. The results also show that the maximum number of jobs that can be treated is 30 for $IP1$, and 70 for $IP2$.

Family-based model ($IP2$)

$$\sum_{m\in M(f)} \sum_{t=1}^{T-p_f+1} x^m_{f,t} = n_f \qquad \forall f \in F \tag{3.11}$$

$$\sum_{m\in M(f)} \sum_{t=1}^{T-p_f+1} (t+p_f-1)x^m_{f,t} \leq C_f \qquad \forall f \in F \tag{3.12}$$

$$\sum_{\tau=t-p_f+1}^{t} x^m_{f,\tau} \leq 1 \qquad \forall t = 1\dots T, \forall f \in F, \forall m \in M(f) \tag{3.13}$$

$$\sum_{\tau=t-p_f-s_{f'}+1}^{t} x^m_{f,\tau} + n_f.x^m_{f',t} \leq n_f \qquad \forall t = 1\dots T, \forall (f,f') \in F\times F \tag{3.14}$$

$$\text{s.t. } f \neq f', \forall m \in M(f) \cap M(f')$$

$$\sum_{\tau=t-\gamma_f+1}^{t} x^m_{f,\tau} + y^m_{f,t} \geq 1 \qquad \forall f \in F, \forall m \in M(f), \forall t = \gamma_f \dots T \tag{3.15}$$

$$y^m_{f,t-1} \leq y^m_{f,t} \qquad \forall t = 2\dots T, \forall f \in F, \forall m \in M(f) \tag{3.16}$$

$$x^m_{f,t} \in \{0,1\} \quad \forall t = 1 \dots T, \forall f \in F, \forall m \in M(f) \qquad (3.17)$$
$$y^m_{f,t} \in \{0,1\} \quad \forall t = 1 \dots T, \forall f \in F, m \in M(f) \qquad (3.18)$$

Constraint (3.11) ensures that n_f jobs are scheduled for family f, and Constraint (3.12) is used to determine C_f. Note that C_f in ($IP2$) is equal to $\sum_{i \in N} C_i$, where $f(i) = f$, and hence Constraints (3.13) and (3.14) are equivalent to Constraints (3.5) and (3.6). The number of binary variables in ($IP2$) is greatly reduced compared to the number of binary variables in ($IP1$). For each family f, the number of binary variables x is divided by n_f.

Evolution of $IP2$

Constraint (3.15) is initially written in our first models as:

$$\sum_{\tau=t-\gamma_f+1}^{t} x^m_{f,\tau} + y^m_{f,t} = 1 \forall f \in F, \forall m \in M(f), \forall t = \gamma_f \dots T. \qquad (3.19)$$

After testing and analyzing the models, it became obvious that this equation is wrong since it forces only one job of family f to be scheduled between $t - \gamma_f + 1$ and t or a qualification run to be done. However, in our hypothesis, at least one job should be scheduled and not only one job. Hence, Constraint (3.15) is now used, where at least one job of a given family is scheduled during an associated time threshold and not necessarily one job.
Moreover, after studying and examining $IP2$, and after a significant period of test and validation of this model, we saw that this model is strongly dependent on the time horizon associated to each test instance. $IP2$ loses machine qualifications even after C_{max}, i.e. the maximum completion time of all jobs, when no jobs are scheduled. More precisely, Constraint (3.16) is dependent on the time horizon ($y^m_{f,t-1} \leq y^m_{f,t}$). To avoid losing a machine qualification, it is necessary to maintain this qualification on the machine from period 1 to T.

However, it is more relevant to consider a model where the number of machine qualification losses at the end of time horizon T, i.e. $\sum_{f \in F} \sum_{m \in M(f)} y^m_{f,T}$, is independent of the time horizon. In order to do this, Constraint (3.16) was first changed to:

$$y^m_{f,t-1} - 1 + \frac{1}{T-(t-1)} \sum_{\tau=t}^{T} \sum_{f' \in F} \sum_{m' \in M(f')} x^{m'}_{f',\tau} \leq y^m_{f,t} \qquad (3.20)$$

$$\forall t = 2 \dots T, \forall f \in F, \forall m \in M(f) \qquad (3.21)$$

It is no longer necessary to maintain a qualification on the machine if no job is started on any machine in the remainder of the horizon, i.e. $\frac{1}{T-(t-1)} \sum_{\tau=t}^{T} \sum_{f' \in F} \sum_{m' \in M(f')} x_{f',\tau}^{m'} = 0$. Hence, the number of machine qualification losses does not depend on T (if T is large enough). Nevertheless, while testing $IP2$ with this new constraint, we saw that, although the number of machine qualification losses is no longer dependent on the time horizon, it still depends on the makespan C_{max}, i.e. the number of machine qualification losses is still being counted between the completion time on a machine and the makespan. To handle this problem, we propose a new version of Constraint (3.16) where Y_f^m is a variable which is equal to 1 if the time threshold is not satisfied for family f on machine m at the end of horizon, and 0 otherwise. Hence, Constraint (3.16) becomes:

$$y_{f,t-1}^m - 1 + \frac{1}{T-(t-1)} \sum_{\tau=t}^{T} \sum_{f' \in F} \sum_{m' \in M(f')} x_{f',\tau}^{m'} \leq Y_f^m \tag{3.22}$$

$$\forall t = 2 \dots T, \forall f \in F, \forall m \in M(f) \tag{3.23}$$

Although only Constraint (3.29) is new in $IP3$ below compared to $IP2$, all other constraints are recalled.

Family-based model ($IP3$)

$$\sum_{m \in M(f)} \sum_{t=1}^{T-p_f+1} x_{f,t}^m = n_f \quad \forall f \in F \tag{3.24}$$

$$\sum_{m \in M(f)} \sum_{t=1}^{T-p_f+1} (t+p_f-1) x_{f,t}^m \leq C_f \quad \forall f \in F \tag{3.25}$$

$$\sum_{\tau=t-p_f+1}^{t} x_{f,\tau}^m \leq 1 \quad \forall t = 1 \dots T, \forall f \in F, \forall m \in M(f) \tag{3.26}$$

$$\sum_{\tau=t-p_f-s_{f'}+1}^{t} x_{f,\tau}^m + n_f . x_{f',t}^m \leq n_f \quad \forall t = 1 \dots T, \forall (f,f') \in F \times F \text{ s.t. } f \neq f', \forall m \in M(f) \cap M(f') \tag{3.27}$$

$$\sum_{\tau=t-\gamma_f+1}^{t} x_{f,\tau}^m + y_{f,t}^m \geq 1 \quad \forall f \in F, \forall m \in M(f), \forall t = \gamma_f \dots T \tag{3.28}$$

$$y_{f,t-1}^m - 1 + \frac{1}{T-(t-1)} \sum_{\tau=t}^{T} \sum_{f' \in F} \sum_{m' \in M(f')} x_{f',\tau}^{m'} \leq Y_f^m \quad \forall t = 2 \dots T, \forall f \in F, \forall m \in M(f) \tag{3.29}$$

$$x_{f,t}^m \in \{0,1\} \quad \forall t = 1 \dots T, \forall f \in F, \forall m \in M(f) \tag{3.30}$$

$$y_{f,t}^m \in \{0,1\} \quad \forall t = 1 \dots T, \forall f \in F, m \in M(f) \tag{3.31}$$

$$Y_f^m \in \{0,1\} \quad \forall f \in F, m \in M(f) \tag{3.32}$$

In Constraint (3.29), it is no longer necessary to maintain a qualification on the machine if no job is started on any machine in the remainder of the horizon, i.e. $\frac{1}{T-(t-1)}\sum_{\tau=t}^{T}\sum_{f'\in F}\sum_{m'\in M(f')} x_{f',\tau}^{m'} = 0$. Hence, the number of machine qualification losses does no longer depend on T (if T is large enough) or on C_{max}. Constraint (3.32) ensures that variable Y_f^m is binary.

Objective function

Two types of criteria can be considered. The first type corresponds to classical scheduling criteria such as minimizing the sum of completion times $\sum_{i\in N} C_i$ (or equivalently $\sum_{f\in F} C_f$ in Models $IP2$ or $IP3$) or the makespan C_{max}. The second type is associated to the number of disqualifications $\sum_{f\in F}\sum_{m\in M(f)} Y_f^m$ (weights could be considered to differentiate between job families w_f or machines w_m). Our objective function is a weighted sum of both types of criteria: $\alpha\sum C_f + \beta\sum Y_f^m$, where α and β are weights that model the trade-off between both criteria. However, it seems more realistic to consider a lexicographical order where the number of qualification runs is prioritized over a pure scheduling criterion, i.e. β is chosen large enough compared to α ($\alpha = 1$, $\beta = |N| * T$), so that improving the scheduling criterion is not preferable to an additional disqualification.

3.5 Scheduling with Equipment Health Factor ($PEHF$)

3.5.1 Motivations

We recall that in chapter 2, the idea of scheduling with Equipment Health Factor (*EHF*) was addressed. In this section, we try to take a further look at the possible approaches that can be used. Although rarely addressed and studied in the literature, we can still find some articles which address EHF. Chen and Wu (2007) develop a new method to determine an effective Preventive Maintenance (PM) scheduling policy based on real-time observations of equipment condition. They use the multivariate process capability index to integrate the equipment multiple parameters into an overall equipment health factor, and then a dynamic PM schedule was determined based on the health prognosis. On the other hand, Guo et al. (1998) presented an integrated real-time equipment monitoring and fault detection approach for chemical vapor deposition tools. This approach includes: simultaneous monitoring scheme (dartboard display of real-time data that provides an easy reading of the equipment overall status), system health factor (index that evaluates the equipment overall health), and analysis functions (including various charting functions, real-time SPC, Run-to-Run SPC, and other advanced SPC functions). Their results show that the proposed system is an effective tool for real-time monitoring and fault detection. Moreover, based on the idea of integrating APC information in scheduling methodologies in semiconductor manufacturing, another possible implementation concerning the usage of EHF is proposed by us. The principle is to assign lot families of different priority/criticality levels, in terms of risk of losing these lots, to machines with various health factors. Figure 3.4 summarizes our idea, where decision of assigning lots to machines with different criticality levels for both, is not necessarily obvious especially when trying to optimize a given risk criterion. We propose that the Equipment Health Factors (EHF) for all machines are defined and known.

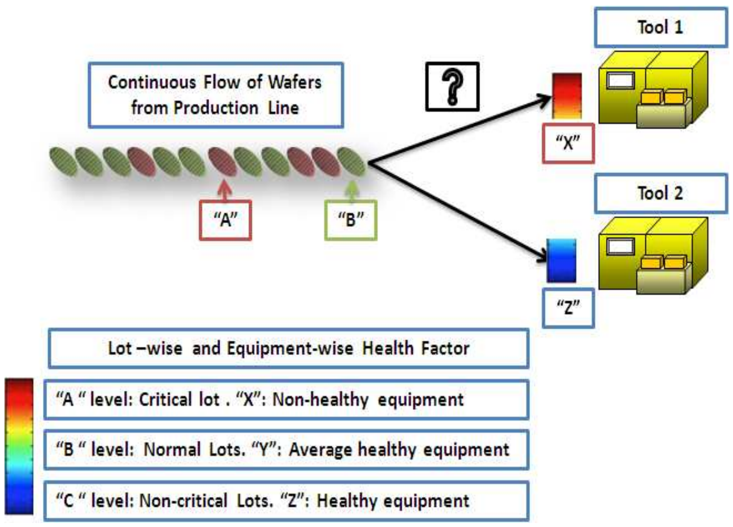

Tool 1
Continuous Flow of Wafers from Production Line
?
"X"
"A"
"B"
Tool 2
"Z"
Lot –wise and Equipment-wise Health Factor
"A " level: Critical lot . "X": Non-healthy equipment
"B " level: Normal Lots. "Y": Average healthy equipment
"C " level: Non-critical Lots. "Z": Healthy equipment

Figure 3.4: Scheduling with EHF

Based on Figure 3.4, two possible *scenarii* of assignment of a lot to machines could be found. In the first scenario, the lot with average risk level is sent to tool 1 with the worst EHF, and the two other high critical lots to tool 2 with a healthy factor. In this case, we may lose the lot sent to tool 1 due to the incompatibility between both factors (lot/machine). This situation is shown in Figure 3.5. Another scenario is also shown where all lots are sent to the equipment with the best EHF. However, by doing this, the makespan will be higher than in the first scenario. On the other hand, the risk of losing lots will be minimized. Hence, we can see the trade-off between these two objective criteria. Therefore, a new interesting field of research would be to look for solutions of this problem.

The problem becomes more and more complex if we take into account

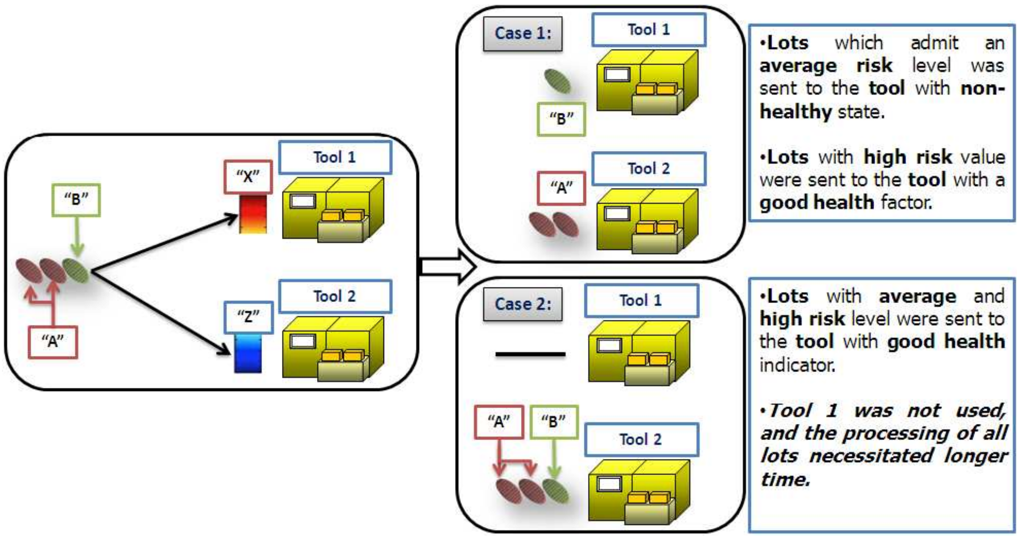

"B"
"X"
Tool 1
"Z"
Tool 2
"A"
Case 1:
Tool 1
"B"
Tool 2
"A"
•Lots which admit an average risk level was sent to the tool with non-healthy state.
•Lots with high risk value were sent to the tool with a good health factor.
Case 2:
Tool 1
"A"
"B"
Tool 2
•Lots with average and high risk level were sent to the tool with good health indicator.
•Tool 1 was not used, and the processing of all lots necessitated longer time.

Figure 3.5: Various lot-to-tool assignment

lot families of different risk levels (criticalities), to be assigned to tools with different EHFs. Figure 3.6 describes the situation where lots of different families with various risk levels should be sent to tools with different EHFs. The decision problem becomes interesting when each assignment/choice is associated to a corresponding risk value, which in its turns could be translated into a cost, a yield and/or a probability of losing a lot/wafer or even chips on this wafer. The question lies in how we present this risk, and what are the possible models which are relevant and efficient. The next section tries to answer these questions.

3.5.2 Risk modeling

Risk is considered as probability and severity in Razzaghi and Kodell (2004). Hubac et al. (2010) study risk assessment and evaluation methods in semiconductor manufacturing. They address the benefits of supporting PM policies by information from APC system. However, They discuss the risk associated to an equipment. In our case, we are looking to model and evaluate the risk related to assigning a job/lot to a tool based on its EHF. We believe this risk can be evaluated in our opinion based on EHF which describes the state of an equipment as a dominant parameter, as well as an indicator associated to the job/lot itself, which describes the severity/criticality level corresponding to the job family.

We propose several ways to model risk, which are shown in Figure 3.7.

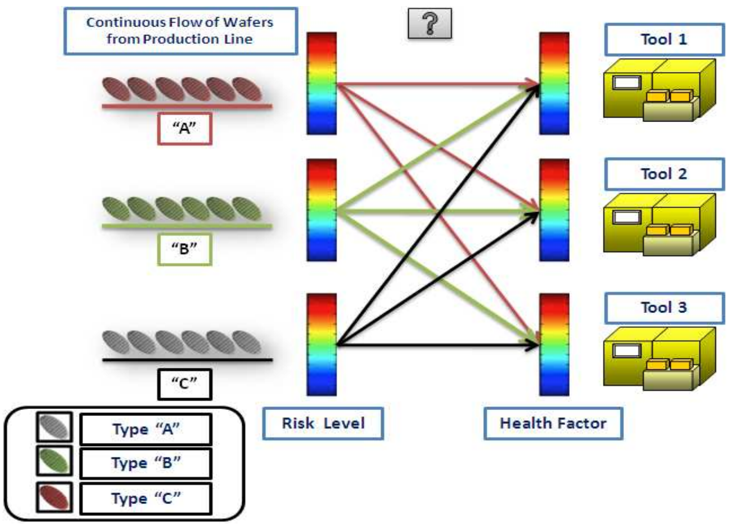


Figure 3.6: General scenario: Various families and different risk levels.

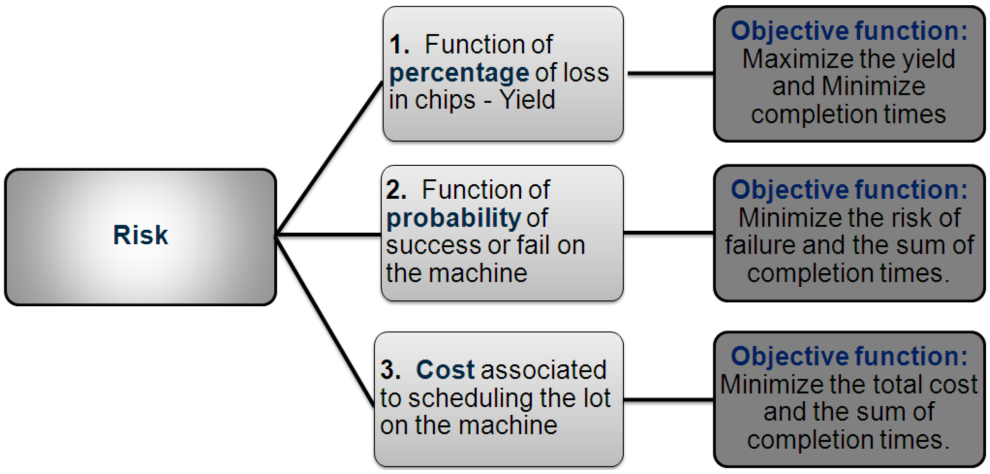

Risk
1. Function of percentage of loss in chips - Yield
2. Function of probability of success or fail on the machine
3. Cost associated to scheduling the lot on the machine
Objective function: Maximize the yield and Minimize completion times
Objective function: Minimize the risk of failure and the sum of completion times.
Objective function: Minimize the total cost and the sum of completion times.

Figure 3.7: Various risk models

1. The risk can be represented as a percentage of loss of chips on a lot. This means the the decision of send a certain job family on a given machine, induces a certain percentage p of loss in chips on this lot/job. This percentage could be predefined in a fixed matrix where the values are fixed for (job family,machine).

2. We may also consider that these values change over time (related to the machine reliability level). This case will not be considered in this thesis. We will assume that the risk of assigning a lot to a given machine is taken from the master matrix of risks that gives the value of risk (percentage of loss, expected yield, etc) which corresponds to the family of each job to each machine. Table 3.2 describes a master matrix, from where risks in their different possible forms corresponding to an assignment of a job family to a machine, can be deducted.

3. Another way to represent the risk of assigning a job to a machine is to consider the probability of loss and not the percentage itself. In both cases, the objective will be to minimize the sum of percentages and/or probabilities of loss.

The values $v(f, m)$ in Table 3.2 can also represent the expected yield that results from an assignment. In this case, the objective will be to maximize the yield while minimizing the sum of completion times and the sum of machine disqualifications. Moreover, we can for sure represent the risk as a cost corresponding to an assignment.

In PTC, when a machine is disqualified for a given job family, it is no longer possible to assign any job of that family. In this context, the new proposed problems of scheduling with EHF extend PTC, by considering that it is no more a matter of complete qualification or disqualification, but a qualification with a certain risk resulting from scheduling job families of a given criticality on full or semi qualified machines. We can look to machine qualifications in this case as levels of reliability and eligibility represented by risk values associated to each job family to machine assignment. In other words, a machine m is considered as qualified to process a job of a certain family f if the risk of associating this job family to this machine $v(f, m)$ is greater than a predefined risk threshold $v_{max}(f, m)$. In what follows, we keep both concepts of machine qualifications and the risk obtained related to an assignment. We assume that, whenever a machine is qualified, lots can be scheduled on it regardless of the risk value, i.e. machines are already disqualified for high risk values or low expected yields. This hypothesis is based on the fact that $PEHF$ is considered as a natural extension of PTC, and to keep on the concept of machine qualifications related to time constraints. Also, keeping on both concepts gives us the flexibility to freely define the value of $v(f, m)$ to be either a yield which should be maximized or a probability of loss to be minimized.

From the reasons above, we decided to use two possible representations of the risk value:

Table 3.2: Family-Machine assignment matrix

	Machine EHF		
Family	X	Y	Z
$F1$	$v(F1, X)$	$v(F1, Y)$	$v(F1, Z)$
$F2$	$v(F2, X)$	$v(F2, Y)$	$v(F2, Z)$
$F3$	$v(F3, X)$	$v(F3, Y)$	$v(F3, Z)$

- $v_{max}(f, m)$ represents the probability of losing the chips or wafers and hence the probability of discarding a lot. This approach of risk is detailed in Section 3.5.4.
- $v_{max}(f, m)$ corresponds to the expected yield resulting from a family to machine assignment. This is more detailed in Section 3.5.5.

3.5.3 Problem definition

We recall that PTC is the problem of scheduling of lots (jobs) of different product types (job family) on parallel machines, where not all machines are able to process all job families (non-identical machines). $PEHF$ could be seen as a normal extension to PTC where, to the time constraint on jobs of the same family, we add the notion of risk of assigning job families to machines. The problem becomes more complicated, since the assignment of jobs to machines is critical to avoid qualification runs on one hand, and on the other, should sometimes be avoided to minimize/maximize the probability of loss/expected yield associated to the assignment.

Consequently, the problem becomes scheduling, on a time horizon discretized in T periods, a set N of jobs of different families on a set M of parallel machines. We recall that the processing times p_f of all jobs in family f are equal and that a setup time s'_f on a machine is necessary to change from one job of a family f to another job of another family f', where $f \neq f'$. Once again, the time constraint is considered through the parameter γ_f, which corresponds to the time threshold. The parameter v_f^m represents in this case the probability of loss. The objective is to optimize a multi-criteria scheduling problem (sum of completion times, sum of losses in machines qualifications, and sum of probability of loss in lots or sum of expected yield). Regarding the sum of probability of loss, we can distinguish two cases:

- Case A. The sum of probability of loss is the sum over all families and machines P^{Max} (independent of families) to be determined. This represents the maximum allowed sum of probability of loss when scheduling all jobs on all machines.

- Case B. The sum of probability is family dependent, i.e. each family admits a P_f^{Max} to be determined. This variable represents the maximum allowed sum of probability of loss for each family f.

3.5.4 First approach: risk as a probability of loss

To represent the problem of scheduling with constraints of probability of loss, the next section addresses the different possibilities of implementation of the notion of risk seen as a probability of loss in our PTC, hence obtaining different possibilities of presenting $PEHF$ in terms of mathematical models.

Mathematical programming model ($IP4$)

Notations We use the same notations that in section 3.4.2. Some new parameters and variables are introduced:

Parameters

v_f^m: Risk value (yield as well as probability of loss) of assigning a job of family f to machine m,
w_f: Weight of family f. In this thesis, we consider that all weights are equal ($w_f = w_{f'}, \forall (f, f') \in F \times F$)

Variables

P_f^{Max}: Maximum sum of probability of loss of jobs per family f,
P^{Max}: Maximum sum of probability of loss of scheduling all jobs on all machines.

Model ($IP4$) below shows the integration of the sum of probability of loss related to machines EHF in PTC (Case A). The objective function in this case should include the notion of probability of loss to be minimized. This objective function is:

$$\min\,(\alpha \sum_{f \in F} C_f + \beta \sum_{f \in F} \sum_{m \in M} Y_f^m + \gamma P^{Max}) \tag{3.33}$$

This insures that the decision of assigning a job family to a certain machine is not only related to the time threshold, but also to the cost related to machines EHF.

Mathematical model with sum of probability of loss ($IP4$)

$$\sum_{m \in M(f)} \sum_{t=1}^{T-p_f+1} x_{f,t}^m = n_f \quad \forall f \in F \tag{3.34}$$

$$\sum_{m \in M(f)} \sum_{t=1}^{T-p_f+1} (t+p_f-1)x_{f,t}^m \leq C_f \quad \forall f \in F \tag{3.35}$$

$$\sum_{\tau=t-p_f+1}^{t} x_{f,\tau}^m \leq 1 \quad \forall t = 1 \dots T, \forall f \in F, \forall m \in M(f) \tag{3.36}$$

$$\sum_{\tau=t-p_f-s_{f'}+1}^{t} x_{f,\tau}^m + n_f . x_{f',t}^m \leq n_f \quad \forall t = 1 \dots T, \forall (f,f') \in F \times F \tag{3.37}$$

$$\text{s.t. } f \neq f', \forall m \in M(f) \cap M(f')$$

$$\sum_{\tau=t-\gamma_f+1}^{t} x_{f,\tau}^m + y_{f,t}^m \geq 1 \quad \forall f \in F, \forall m \in M(f), \forall t = \gamma_f \dots T \tag{3.38}$$

$$y_{f,t-1}^m - 1 + \frac{1}{T-(t-1)} \sum_{\tau=t}^{T} \sum_{f' \in F} \sum_{m' \in M(f')} x_{f',\tau}^{m'} \leq Y_f^m \quad \forall t = 2 \dots T, \forall f \in F, \forall m \in M(f) \tag{3.39}$$

$$\mathbf{\sum_{f \in F} \sum_{m \in M(f)} \sum_{t=1}^{T} x_{f,t}^m v_f^m \leq P^{Max}} \tag{3.40}$$

$$x_{f,t}^m \in \{0,1\} \quad \forall t = 1 \dots T, \forall f \in F, \forall m \in M(f) \tag{3.41}$$

$$y_{f,t}^m \in \{0,1\} \quad \forall t = 1 \dots T, \forall f \in F, m \in M(f) \tag{3.42}$$

$$Y_f^m \in \{0,1\} \quad \forall f \in F, m \in M(f) \tag{3.43}$$

$$P^{Max} \geq 0 \tag{3.44}$$

Constraint (3.40) helps to determine the maximum allowed possible total sum of probability of loss resulting from the assignment of job families to machines and its associated cost ($\sum_{m \in M(f)} \sum_{t=1}^{T} x_{f,t}^m v_f^m$).
Case B considers the maximum allowed sum of probability of loss per family f (P_f^{Max}). This is done by modifying Constraint (3.40) to become:

$$\sum_{m \in M(f)} \sum_{t=1}^{T} x_{f,t}^m v_f^m \leq P_f^{Max}, \forall f \in F \tag{3.45}$$

Consequently, the objective function becomes:

$$\min \left(\alpha \sum_{f \in F} C_f + \beta \sum_{f \in F} \sum_{m \in M} Y_f^m + \gamma \sum_{f \in F} w_f P_f^{Max}\right) \tag{3.46}$$

3.5.5 Second approach: maximizing yield as an objective

In this case, the value of v_f^m is considered as the expected yield resulting from an assignment of a job of family f to a machine m. The objective function includes three criteria which are the sum of completion times, the number of machine disqualifications and the sum of the expected yield. We propose a weighted sum of these three criteria in our objective function. Equation (3.47) represents the objective function to be maximized with the same constraints that model $IP3$. $IP5$ corresponds to $IP3$ with the next objective function:

$$\max\,(\gamma^{'} \sum_{f\in F}\sum_{m\in M}\sum_{t=1}^{T} w_f x_{f,t}^m v_f^m - (\alpha^{'} \sum_{f\in F} C_f + \beta^{'} \sum_{f\in F}\sum_{m\in M} Y_f^m)) \tag{3.47}$$

3.6 Conclusion

In this chapter, we have discussed two new problems which address the integration of scheduling and APC. Concerning the scheduling aspect, these problems are the scheduling of job families on non-identical parallel machines under special constraints, i.e. with time constraints in PTC, and with EHF in $PEHF$. Simulation on test instances for exact methods and further analysis are elaborated in the next chapter (Chapter 4). Development of solution approaches (heuristics and metaheuristics) are addressed in Chapter 5.

Chapter 4

Numerical experiments on Mathematical Programming Models

4.1 Introduction

The problem of scheduling with APC constraints represented by a time constraint (PTC) is numerically analyzed in this chapter. Also, the results of simulations on the problem of scheduling with Equipment Health Factor ($PEHF$) are shown. The tests are conducted on randomly generated instances by an instance generator that was developed for this reason.

Models ($IP1$, $IP2$, $IP3$ and $IP5$) are tested using a standard solver (Xpress-MP), on an Intel Xeon processor of 2.50 GHz and 3 GB of RAM. Exact solutions are obtained for several types of instances that were generated with 10, 20, 30, 40, 50, 60, 70 and 80 jobs, 2, 3, 4 and 5 families, 2, 3, 4 and 5 machines, and upper bounds on processing times of 10, 20 and 30. The objective function is considered as a weighted combination of different criteria ($\alpha \sum C_f + \beta \sum Y_f^m$ for PTC and $\gamma' \sum_{f \in F} \sum_{m \in M} \sum_{t=1}^{T} w_f x_{f,t}^m v_f^m - (\alpha' \sum_{f \in F} C_f + \beta' \sum_{f \in F} \sum_{m \in M} Y_f^m))$ for $PEHF$). First, the values of the weights are varied in order to study their effect on each criterion of the objective function (determination by linear combination of criteria). Examples on an ϵ-constraint Pareto optima method are also provided. In addition, the weights are considered to prioritize one criterion over other criteria in a lexicographical order. Let us recall that multicriteria optimization in scheduling and the choice of the objective function is addressed in Chapter 3. The execution time for all models and each instance are limited to 600 seconds.

The rest of this chapter is organized as follows. In Section 4.2, instance generation characteristics are described. Section 4.3 provides the obtained numerical values of exact methods when solving models $IP1$, $IP2$ and $IP3$ for PTC and $IP5$ for problem $PEHF$. Both problems are analyzed in Section 4.4 regarding their objective functions (PTC and $PEHF$), impact of threshold variations and Pareto frontiers (PTC). Section 4.5 concludes this chapter.

4.2 Instances generation and characteristics

4.2.1 Instance generation

To test the mathematical programming models for problems PTC and $PEHF$ and the heuristics in Chapter 5, test instances were randomly generated. The different parameter values to generate the instances were chosen so that the basic problem pre-requisites are respected.
For both problems, the time thresholds of job families were set sufficiently large with respect to their associated processing times. This was done to give a minimal bias to find a solution, since short thresholds may lead to a very fast loss in machine qualifications. Hence, it will not be possible to process all available jobs, since jobs cannot be sent to disqualified machines based on our hypothesis. We consider that $Max(p_f) \leq Min(\gamma_f)$. The initial family/machine qualification scheme was defined so that each family has at least one machine on which it can be processed, and each machine is qualified to process at least one job family. This is to inhibit a direct infeasible solution of a given instance. Setup times were not chosen too large so as the probability of losing a machine qualification due to a set-up time insertion is acceptable. We consider that $Max(s_f) \leq Min(p_f)$. In addition, the time horizon was taken as the sum of all processing times, plus the setup time multiplied by the number of jobs per family. This is an extreme case where all jobs are scheduled on one single machine and where, each time a job is scheduled, a setup time is required ($T = \sum_{f \in F} n_f * (p_f + s_f)$). The instances/instance types were chosen to consider the possible effects of the number of jobs, families, machines and also the upper bound of the family processing times.
In addition, the value of v_f^m i.e., the expected yield resulting from an assignment of a job of family f to a machine m, is considered to be dominant per machine. This means that, whenever a machine m^* is *healthier* than

m', then the yield obtained when assigning any job family f to m^* is better than when assigning the same job family f to any other machine m', i.e. $v_f^{m^*} \geq v_f^{m'}, \forall f \in F$. This hypothesis is strong, it prioritizes the Equipment Health Factor on any other factor as for example the criticality of the processed family/lot as noted in Chapter 3. We believe that this hypothesis added to the existence of the time threshold γ_f related to job families, add more importance to EHF by making the consequence of a scheduling decision (assigning a machine to a job family) more machine based/dependent. On the other hand, the notion of the lot/wafer family criticality index (LCI) is still considered while defining the values of the resulting yield v_f^m where $v_f^m = f(EHF, LCI)$.

4.2.2 Instance characteristics

In the sequel, the studied instances represent the types of instances. Each type is a group of 10 instances, the values of the parameters are generated following the Gaussian law, and the results are taken as an average of these instances. In addition, the mean values of the sum of completion times and those of the yield are rounded up to the nearest integer since they are sufficiently large to notice the difference when comparing them for different objective functions. The mean values of the number of setups are also rounded up to the nearest integer since it is not a direct objective and it is related to the sum of completion times. Moreover, the percentage of obtained optimum solutions (over 10 instances), for each instance type, is also provided. The average value of the sum of losses in machine qualifications, also found in the sequel to be the total number of machine disqualifications, is also provided.

Table 4.1 provides the characteristics of the tested instance types. In the first column, each instance type is indexed by a number (*No.*). These representative instance types summarize different possibilities of combinations between the number of jobs $|N|$ (second column), number of machines $|M|$ (third column) and number of families $|F|$ (fourth column). The bounds on the processing times are of a remarkable importance since the time horizon T (column 8) is a function of this parameter. While solving a time indexed mixed integer linear programming model, the resolution time is dependent on the given time horizon and hence on these predefined bounds. In column 5, "Max.(p_f)" represents the upper bound of the generated processing times of all families in a given instance ($p_f, \forall f \in F$). The maximum possible sum of machine-family qualifications $|M| * |F|$ is given in Column 6. The initial sum of qualified machines for all families $\sum_f |M_o(f)|$ of a given instance is provided in Column 7. This value helps in knowing the percentage of the initial sum of qualified machines for all families with respect to the maximum possible number of qualified machines (all machines are qualified to process all families). This value is rounded up to nearest the integer.

4.3 Solving the mathematical programming models

In this section, we solve the different models of both problems. It is important to mention before that classical multi-objective optimization methods are usually classified into four classes: No-preference methods, Posteriori methods, A priori methods and Interactive methods (Miettinen (1999)). The no preference methods do not assume any information on the importance of objectives. These methods do not make any attempt to find multiple Pareto-optimal solutions. Posteriori methods use preference information of each objective and iteratively generate a set of Pareto-optimal solutions. However, generating Pareto-optimal solutions require some knowledge on algorithmic parameters which will ensure finding a Pareto-optimal solution. A priori methods use more information about the preferences of objectives and usually find one preferred Pareto-optimal solution. Interactive methods use the preference information progressively during the optimization process. In the next section, the weighted sum method is used. This method scalarizes the set of objectives ($\sum_{f\in F} C_f$, $\sum_f \sum_m y_f^m$ and/or $\sum_{t=1}^{T} \sum_f \sum_m x_{f,t}^m v_{f,t}^m$) into a single objective by pre-multiplying each objective with a predefined weight (this weight is usually defined by the decision maker). This method is probably one of the most widely used classical approaches. Faced with multiple objectives, this method is the most convenient one that comes to mind. For example, regarding the two objectives of minimizing the sum of completion times and minimizing the sum of losses in machine qualifications, one naturally thinks of minimizing a weighted sum of these two objectives.

Although the idea is not so complicated, it introduces an important question. Which values of the weights must one use? Actually, there is no unique answer to this question. The answer depends on the importance of each objective in the context of the problem and the scaling factors associated with the objective function (e.g. $\alpha = 1$, $\beta = |N| * T$ and $\gamma = |M| * |f| * |N| * T$). Different objectives take different orders of magnitude (e.g. $\sum_{f\in F} C_f$ of 1000 and $\sum_f \sum_m y_f^m$ of 10). Hence when such objectives are weighted to form a composite objective function, it would be better to scale them appropriately

Table 4.1: Instance characteristics

$No.$	$\|N\|$	$\|M\|$	$\|F\|$	Max.(p_f)	$\|M\| * \|F\|$	$\sum_f (\|M_o(f)\|)$	T
1	10	2	2	10	4	3	85
2	10	2	2	20	4	3	90
3	10	2	2	30	4	3	168
4	10	2	3	10	6	5	74
5	10	2	3	20	6	5	108
6	10	2	3	30	6	5	150
7	10	3	3	10	9	7	56
8	10	4	4	10	16	10	44
9	10	5	2	10	10	6	70
10	10	6	2	10	12	8	32
11	10	7	2	10	14	8	66
12	10	8	2	10	16	10	56
13	10	9	2	10	18	10	32
14	20	3	4	10	12	8	101
15	20	3	5	10	15	9	60
16	20	4	2	10	8	6	132
17	20	4	3	10	12	7	55
18	20	4	4	10	16	11	134
19	20	4	5	10	20	13	114
20	30	3	2	10	6	4	210
21	30	3	3	10	9	7	124
22	30	3	4	10	12	8	95
23	30	3	5	10	15	13	118
24	30	4	4	10	16	10	162
25	30	5	5	10	25	14	144
26	40	3	2	10	6	4	175
27	40	3	3	10	9	5	284
28	40	3	4	10	12	7	170
29	40	3	5	10	15	10	252
30	40	4	4	10	16	11	264
31	50	2	2	10	4	3	225
32	50	2	3	10	6	4	247
33	50	2	4	10	8	7	243
34	50	3	3	10	9	6	299
35	50	5	2	10	10	8	356
36	50	5	3	10	15	9	237
37	50	5	4	10	20	15	249
38	50	5	5	10	25	14	193
39	60	3	4	10	12	9	348
40	60	3	5	10	15	11	289
41	60	5	4	10	20	14	201
42	60	5	5	10	25	13	160
43	70	2	4	10	8	6	259
44	70	3	5	10	15	12	283
45	70	4	4	10	16	9	329
46	70	4	5	10	20	14	398
47	70	5	4	10	20	12	474
48	70	5	5	10	25	16	341
49	80	2	2	10	4	3	532
50	80	2	3	10	6	5	497

so that each has more or less the same order of magnitude. this process is called *normalization* of objectives. The weight of an objective is usually chosen in proportion to the objective relative importance in the problem (PTC or $PEHF$).

4.3.1 Models $IP1$, $IP2$ and $IP3$ (PTC)

In this section, numerical results obtained with Models $IP1$, $IP2$ and $IP3$ are shown. In the following tables below, columns named $\sum_{j\in J} C_j$ and $\sum_f C_f$ indicate the obtained sum of completion times for jobs/families respectively. Columns named $\sum_f \sum_m y^m_{f,T}$ and $\sum_f \sum_m Y^m_f$ indicate the sum of loss in machine qualifications for time horizon dependent and independent respectively. Columns denoted by *Setups* indicate the total number of setups needed in the solution of the instance type. CPU-T is the CPU time to solve the instance type.

The two weights of both criteria α and β, let us call them the preference vector (α,β), are an important issue in our analysis of the objective function and/or the problem itself. First, in the multicriteria objective function of both problems (PTC and $PEHF$), we consider a weighted sum of these criteria, i.e. a preference vector of different values, to solve Models $IP1$, $IP2$, $IP3$ and $IP5$. This is usually named as a preference-based multi-objective optimization method. Nevertheless and generally speaking, it is important to realize that the trade-off solution obtained by using the preference-based strategy is sensitive to the relative preference vector used in the objective function ($\alpha = 1$, $\beta = |N| * T$). In the next sections, the obtained results show that a change in this preference vector results in a different trade-off solution which is not always true for all types of multi-objective optimization problems. In fact, it is intuitive to realize that finding a relative preference vector itself is highly subjective and not straightforward (Deb (2001)). This requires an analysis of the non-technical, qualitative and experience-driven information to find a quantitative relative preference vector. Without any knowledge of the likely trade-off solutions, this is an even more difficult task. Consequently, both problems are analyzed in Section 4.4 to prove the existing trade-off and to study the effect of choosing different preference vectors.

Solving Model $IP1$

Table 4.2 provides the obtained values on various test instance types, of the sum of completion times and the sum of losses in machine qualifications for exact solutions using Model $IP1$. It shows the results when both criteria of the objective function are optimized, using the following values: $\alpha = 1$ and $\beta = 1$ (i.e. $\sum_{j \in J} C_j + \sum_{f \in F} \sum_{m \in M(f)} y^m_{f,T}$). Taking these values for α and β means that minimizing the sum of completion times is prioritized over the sum of losses in machine qualifications.

The execution times (values in CPU-T column) increase when the upper bound on the processing time increases as it can be seen in the groups of instance types (1, 2 and 3) and (4, 5 and 6) which have the same number of jobs, families and machines but different upper bounds on job family processing times (10, 20 and 30 respectively). This is an expected behavior since the CPU time of a time indexed MILP model is time horizon dependent ($T = \sum_{f \in F} n_f * (p_f + s_f)$). In other words, when processing times become larger, the time horizon increases as well and, in our time indexed models, the number of variables is directly related to the number of periods in the horizon. As expected, the number of jobs also impacts resolution times. For example, the execution times in instance types (1, 31 and 49), (4, 32 and 50), (7, 21, 27 and 34), (8, 18, 24 and 30) and (9 and 35) increase while fixing the number of families and machines, the upper bound on processing times, and varying the number of scheduled jobs. Note that Instance types 34 to 50 admit no solution with Model $IP1$ because of an *out of memory* of the solver. In addition, no optimal solutions are found for Instance types 17 to 33 because the solver attained the time limit of 600 seconds. The best feasible solution is kept. Also, the number of families and machines has an equivalent effect on the CPU time and hence on the complexity of the resolution of an instance. For example, Instance types (9, 11 and 12), respectively (1 and 4), the CPU time is increased whenever the number of machines, respectively families is increased while fixing other parameters (number of jobs and processing time upper bound).

However, the resolution times of Instance types 9, 10 and 13 show that other parameters of the problem, and more specifically the initial qualification scheme (6, 8 and 10 respectively), also has an effect on the problem complexity and the CPU time, where the instance with the largest number of machines takes less time to be solved. Actually, the larger the number of machines initially qualified to process job families, the easier the instance usually is. In addition, the thresholds associated to each job family also impact the complexity of problems, since short thresholds may lead to fast machine disqualifications. Hence, it becomes more difficult to find a solution in which all job families are scheduled.

Moreover, the maximum number of scheduled jobs below the time limits is found to be 20 jobs for Model $IP1$ while prioritizing the sum of completion times of jobs of different families. Instance types with more than 50 jobs (Instance types 33 to 50) cannot be solved under the given testing environment (*out of memory*). In the next section, the results obtained from Model $IP2$ on the same instances are shown and compared to those of Model $IP1$.

Table 4.2: Model $IP1$, ($\alpha = 1$, $\beta = 1$), Minimizing the sum of completion times and the number of losses in machine qualifications

$No.$	$\sum_{j\in J}$ (C_j)	$\sum_f \sum_m$ $(y^m_{f,T})$	Setups	CPU	% of Opt.	No.	$\sum_{j\in J}$ (C_j)	$\sum_f \sum_m$ $(y^m_{f,T})$	Setups	CPU-T	% of Opt.
1	255	0.1	2	3.1	100 %	26	1108	0.4	5	600	0 %
2	245	1.4	3	13.7	100 %	27	2256	2.3	4	600	0 %
3	564	2.3	2	14.6	100 %	28	1228	4.2	6	600	0 %
4	211	4.9	3	6.4	100 %	29	1854	9.9	5	600	0 %
5	278	4.8	3	35.5	100 %	30	1705	8.9	9	600	0 %
6	351	4.9	3	85.5	100 %	31	3447	2.4	5	600	0 %
7	124	4.7	4	4.5	100 %	32	3343	2.7	4	600	0 %
8	76	9.8	6	1.2	100 %	33	3203	2.9	8	600	0 %
9	116	3.9	5	0.9	100 %	34	-	-	-	-	
10	46	4.6	6	0.6	100 %	35	-	-	-	-	
11	96	7.7	6	1	100 %	36	-	-	-	-	
12	67	9.8	8	1.1	100 %	37	-	-	-	-	
13	34	9.9	9	0.2	100 %	38	-	-	-	-	
14	346	7.8	7	600	20 %	39	-	-	-	-	
15	285	6.7	8	11.3	100 %	40	-	-	-	-	
16	404	0.4	7	600	20 %	41	-	-	-	-	
17	157	5.8	7	4.7	100 %	42	-	-	-	-	
18	407	10.9	8	600	20 %	43	-	-	-	-	
19	299	12.6	8	600	10 %	44	-	-	-	-	
20	1177	0.4	5	600	20 %	45	-	-	-	-	
21	582	4.8	7	600	10 %	46	-	-	-	-	
22	521	7.9	8	600	10 %	47	-	-	-	-	
23	642	12.8	10	600	0 %	48	-	-	-	-	
24	630	6.2	9	600	10 %	49	-	-	-	-	
25	463	13.9	11	600	0 %	50	-	-	-	-	

Table 4.3 shows the results when both criteria of the objective function are optimized, using the following values: $\alpha = 1$ and $\beta = |N| * T$. These values are chosen so that the scheduling criterion, minimizing the sum of completion times, is second to the minimization of the number of disqualifications, so that it is equivalent to a lexicographical order. This table also provides the values of the sum of completion times and the sum of losses in machine qualifications for solutions obtained using Model $IP1$.

The exact solutions for both criteria shown in both tables (Table 4.2 and Table 4.3) prove that, when the number of losses in machine qualifications is prioritized (Table 4.3), the sum of completion times is not minimized. Furthermore, when the sum of completion times is prioritized over the sum of losses in machine qualifications (Table 4.2), the results show that the sum of losses in machine qualifications is not minimized. the number of setups while prioritizing the sum of losses in machine qualifications in Table 4.3 is greater than, for all instances, those of Table 4.2. This is due to the fact that job families have to change frequently in order to satisfy the time threshold constraints, hence minimizing $\sum_{f\in F}\sum_{m\in M}(y^m_{f,T})$.

Note that instance types 32 and 33 which have a solution in Table 4.2 are no longer feasible under the same testing environment conditions in Table 4.3. This illustrates that the complexity of the problem is also based on the chosen objective function, where prioritizing the qualification criterion over the scheduling one makes instances more difficult to solve.

Table 4.3: Model $IP1$, ($\alpha = 1$, $\beta = |N| * T$), Minimizing the sum of completion times and the number of losses in machine qualifications

$No.$	$\sum_{j \in J}$ (C_j)	$\sum_f \sum_m$ $(y^m_{f,T})$	Setups	CPU	% of Opt.	No.	$\sum_{j \in J}$ (C_j)	$\sum_f \sum_m$ $(y^m_{f,T})$	Setups	CPU-T	% of Opt.
1	255	0.0	2	7	100 %	26	1108	0.4	6	600	0 %
2	245	0.5	3	10.7	100 %	27	2780	0.3	15	600	0 %
3	830	0.5	5	4.7	100 %	28	1448	0.4	13	600	0 %
4	269	2.6	7	9.3	100 %	29	2363	5.6	15	600	0 %
5	391	2.7	6	600	10 %	30	2094	0.4	16	600	0 %
6	634	1.6	7	40.40	100 %	31	2403	0.4	3	597	20 %
7	185	2.5	7	1.4	100 %	32	-	-	-	-	
8	161	6.7	10	3.2	100 %	33	-	-	-	-	
9	196	2.7	7	4.1	100 %	34	-	-	-	-	
10	102	3.5	7	0.5	100 %	35	-	-	-	-	
11	208	4.6	6	0.8	100 %	36	-	-	-	-	
12	290	5.7	10	14.2	100 %	37	-	-	-	-	
13	148	7.8	9	1.2	100 %	38	-	-	-	-	
14	826	2.8	17	41.8	100 %	39	-	-	-	-	
15	285	6.7	8	30.4	100 %	40	-	-	-	-	
16	404	0.2	9	600	0 %	41	-	-	-	-	
17	397	1.9	16	8.1	100 %	42	-	-	-	-	
18	793	4.9	14	600	0 %	43	-	-	-	-	
19	477	9.9	11	600	0 %	44	-	-	-	-	
20	1177	0.4	7	600	0 %	45	-	-	-	-	
21	682	1.8	9	600	0 %	46	-	-	-	-	
22	802	2.1	20	600	0 %	47	-	-	-	-	
23	806	8.9	18	600	0 %	48	-	-	-	-	
24	918	2.7	13	600	0 %	49	-	-	-	-	
25	1341	7.8	23	600	0 %	50	-	-	-	-	

In the following sections, a graphical interpretation of the results are shown for 9 selected instances (Instance types: 14, 19, 21, 22, 25, 38, 39, 41 and 44). In fact, these instance types are chosen to be representatives of all instance types. This helps in making figures more comprehensible. However, the numerical results for all instance types are given in the corresponding tables.

Model $IP1$ versus Model $IP2$

Let us recall from Chapter 3 that after analyzing the problem and the results obtained with Model $IP1$, it is possible to see that, since there are no release dates on jobs and jobs of the same family have the same processing time, all jobs in a family can be interchanged in an optimal solution. Thereafter, the job variables in Model $IP1$ are grouped into families in Model $IP2$.

Consequently, numerical tests are conducted to show that the family based model $IP2$ dominates the job based model $IP1$ in terms of execution times. Tables 4.4 and 4.5 show the results of the tests conducted on both models. The sum of completion times is now denoted by $\sum_f C_f$ in these tables. It is equivalent to $\sum_{j \in J} C_j$ in Model $IP1$.

The execution times for Model $IP1$ are equal on average to hundred times the execution times for Model $IP2$. Model $IP1$ cannot optimally solve instances with more than 20 jobs when prioritizing the sum of losses in machine qualifications, whereas Model $IP2$ can solve instance up to 70 jobs (Tables 4.3 and 4.5). This shows that Model $IP2$ dominates Model $IP1$, which was expected due to the aggregation of variables.

Both models obtain the same results for the same studied objective function and below the time limits, regarding the sum of completion times in Table 4.4 and the sum of losses in machine qualifications in Table 4.5. Results obtained for instances with CPU time equals to 600 seconds cannot be compared since it is not guaranteed that the solver found an optimum solution. Once again, regarding the prioritized criterion in the objective function, it is found that the resolution of the studied instances in both models ($IP1$ and $IP2$) takes longer CPU times when prioritizing the sum of loss in machine qualifications ($\alpha = 1$, $\beta = |N| * T$) compared to prioritizing the sum of completion times ($\alpha = 1$, $\beta = 1$), see Tables 4.4 and 4.5. Also, this is also true for the increase of the CPU time as a function of number of jobs, families and machines.

Table 4.4: Models $IP1$ and $IP2$, ($\alpha = 1$, $\beta = 1$), Minimizing the sum of completion times and the number of losses in machine qualifications

	$\sum_f C_f$		$\sum_f \sum_m y^m_{f,T}$		Setups		CPU		% of Opt.	
No.	IP1	IP2	IP1	IP2	IP1	IP2	IP1	IP2	IP1	IP2
1	255	255	0.1	0.1	2	2	3.1	0.2	100 %	100 %
2	245	245	1.4	1.4	3	3	13.7	0.3	100 %	100 %
3	564	564	2.3	2.3	2	2	14.7	0.2	100 %	100 %
4	211	211	4.9	4.9	3	3	6.5	0.8	100 %	100 %
5	278	278	4.8	4.8	3	3	35.6	2.9	100 %	100 %
6	351	351	4.9	4.9	3	3	85.5	6.5	100 %	100 %
7	124	124	4.7	4.7	4	4	4.5	1.1	100 %	100 %
8	76	76	9.8	9.8	6	6	1.3	0.6	100 %	100 %
9	116	116	3.9	3.9	5	5	1	0.1	100 %	100 %
10	46	46	4.6	4.6	6	6	0.6	0	100 %	100 %
11	96	96	7.7	7.7	6	6	1	0.1	100 %	100 %
12	67	67	9.8	9.8	8	8	1.2	0.1	100 %	100 %
13	34	34	9.9	9.9	9	9	0.3	0	100 %	100 %
14	346	346	7.8	7.6	7	5	600	3.5	20 %	100 %
15	285	285	6.7	6.7	8	8	11.4	2.7	100 %	100 %
16	404	404	0.4	0.3	7	6	600	2.4	20 %	100 %
17	157	157	5.8	5.8	7	7	4.8	1.2	100 %	100 %
18	407	399	10.9	10.8	8	7	600	8.4	20 %	100 %
19	299	297	12.6	12.6	8	6	600	9.9	10 %	100 %
20	1177	1163	0.4	0.3	5	4	600	2.3	20 %	100 %
21	582	559	4.8	4.4	7	6	600	2.9	10 %	100 %
22	521	483	7.9	7.6	8	6	600	4.4	10 %	100 %
23	642	538	12.8	12.7	10	7	600	17.5	0 %	100 %
24	630	621	6.2	6.0	9	7	600	12.9	10 %	100 %
25	463	427	13.9	13.8	11	11	600	5.1	0 %	100 %
26	1108	1108	0.4	0.3	5	4	600	0.5	0 %	100 %
27	2256	2010	2.3	2.1	4	4	600	2.8	0 %	100 %
28	1228	1117	4.2	4.1	6	5	600	2.5	0 %	100 %
29	1854	1608	9.9	9.8	5	3	600	52.6	0 %	100 %
30	1705	1355	8.9	6.8	9	8	600	38.2	0 %	100 %
31	3447	2394	2.4	0.4	5	3	600	0.5	0 %	100 %
32	3343	3243	2.7	2.3	4	3	600	1.5	0 %	100 %
33	3203	2621	2.9	2.4	8	6	600	31.1	0 %	100 %
34	-	2710	-	1.8	-	5	-	10.6	-	100 %
35	-	1926	-	0.3	-	7	-	6.9	-	100 %
36	-	1329	-	6.8	-	6	-	6.6	-	100 %
37	-	1294	-	8.8	-	8	-	51.5	-	100 %
38	-	962	-	13.9	-	10	-	5.8	-	100 %
39	-	2992	-	4.8	-	7	-	23.3	-	100 %
40	-	2504	-	9.7	-	8	-	33.6	-	100 %
41	-	1158	-	9.9	-	11	-	11.1	-	100 %
42	-	834	-	8.7	-	22	-	6.6	-	100 %
43	-	3626	-	4.9	-	4	-	8.7	-	100 %
44	-	2711	-	7.9	-	9	-	32.1	-	100 %
45	-	2603	-	4.8	-	6	-	6.8	-	100 %
46	-	3279	-	6.8	-	8	-	600	-	20 %
47	-	3429	-	8.9	-	7	-	68.3	-	100 %
48	-	2387	-	11.8	-	12	-	97	-	100 %
49	-	-	-	-	-	-	-	-	-	-
50	-	-	-	-	-	-	-	-	-	-

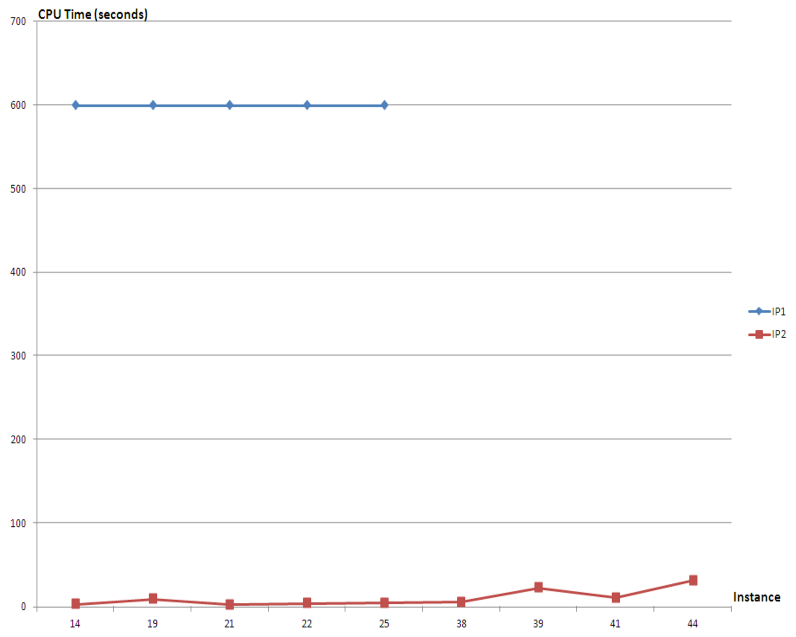


Figure 4.1: Model $IP1$ vs. Model $IP2$, ($\alpha = 1$, $\beta = 1$), CPU times

Figures 4.1 and 4.2 represent the variation of CPU times on the selected instances for both models (Model $IP1$ and Model $IP2$), and for two different objective functions. In Figure 4.1, where the sum of completion times is prioritized over machine qualifications, the CPU times of Model $IP1$ exceed those of Model $IP2$ for all instance types. The same behavior is noticed in Figure 4.2 where machine qualifications are prioritized over the sum of completion times. This is an expected behavior because in Model $IP2$, the number of variables related to the number of jobs is decreased to the number of job families.

Model $IP2$ versus Model $IP3$

Let us recall that the difference between Models $IP2$ and $IP3$ (introduced in Chapter 3) lies in two points. First, in model $IP2$, the calculation of the losses in qualifications depends on the given time horizon $(\sum_{f\in F}\sum_{m\in M(f)} y^m_{f,T})$. This means that, for the larger time horizons, all machines could lose their qualifications. Hence, comparison between different methodologies for solving PTC is not realistic and even solutions cannot be compared. Second, in an enhanced version of model $IP2$ where the goal was to eliminate the time horizon dependency on the calculation of losses in machine qualifications

Table 4.5: Models $IP1$ and $IP2$, ($\alpha = 1$, $\beta = |N| * T$), Minimizing the sum of completion times and the number of losses in machine qualifications

	$\sum_f C_f$		$\sum_f \sum_m y^m_{f,T}$		Setups		CPU-T		% of Opt.	
$No.$	IP1	IP2	IP1	IP2	IP1	IP2	IP1	IP2	IP1	IP2
1	255	255	0.0	0.0	2	2	7.1	0.2	100 %	100 %
2	245	245	0.5	0.5	3	3	10.8	0.7	100 %	100 %
3	830	810	0.5	0.5	6	5	4.8	0.2	100 %	100 %
4	269	249	2.6	2.6	9	7	9.4	1.8	100 %	100 %
5	391	298	2.7	2.7	7	5	600	15.1	10 %	100 %
6	634	500	1.6	1.6	8	7	40.4	5.5	100 %	100 %
7	185	176	2.5	2.5	8	7	1.5	0.2	100 %	100 %
8	161	143	6.7	6.7	11	10	3.3	0.2	100 %	100 %
9	196	184	2.7	2.7	9	7	4.2	0.4	100 %	100 %
10	102	98	3.5	3.5	8	7	0.6	0	100 %	100 %
11	208	174	4.6	4.6	7	6	0.9	0.5	100 %	100 %
12	290	248	5.7	5.7	13	10	14.2	0.1	100 %	100 %
13	148	132	7.8	7.8	10	9	1.3	0.8	100 %	100 %
14	826	782	2.8	2.8	20	17	41.8	2.9	100 %	100 %
15	285	285	6.7	6.7	8	8	30.4	3.3	100 %	100 %
16	404	404	0.2	0.2	6	6	600	2.4	0 %	100 %
17	397	381	1.9	1.9	17	16	8.2	0.6	100 %	100 %
18	793	739	4.9	4.6	13	13	600	5.8	0 %	100 %
19	477	441	9.9	9.6	10	9	600	9.7	0 %	100 %
20	1177	1163	0.4	0.3	5	4	600	2.6	0 %	100 %
21	682	634	1.8	1.7	9	8	600	7.3	0 %	100 %
22	802	841	2.1	0.6	19	20	600	2.6	0 %	100 %
23	806	732	8.9	8.7	18	16	600	600	0 %	100 %
24	918	862	2.7	2.7	14	12	600	13.5	0 %	100 %
25	1341	1272	7.8	7.6	24	23	600	5	0 %	100 %
26	1108	1108	0.4	0.3	4	4	600	0.3	0 %	100 %
27	2780	2614	0.3	0.2	15	13	600	41.5	0 %	100 %
28	1448	1370	0.4	0.3	14	12	600	6.7	0 %	100 %
29	2363	1972	5.6	5.0	17	14	600	275.5	0 %	90 %
30	2094	1936	0.4	0.3	17	16	600	41.2	0 %	100 %
31	2403	2394	0.4	0.3	5	3	597	0.7	20 %	100 %
32	-	3837	-	0.2	-	9	-	103.1	-	100 %
33	-	3293	-	0.2	-	20	-	151.5	-	100 %
34	-	4033	-	0.3	-	17	-	48.2	-	100 %
35	-	1926	-	0.3	-	7	-	6.9	-	100 %
36	-	2686	-	0.9	-	23	-	1.5	-	100 %
37	-	1635	-	4.8	-	16	-	600	-	10 %
38	-	2602	-	5.8	-	31	-	8.1	-	100 %
39	-	4361	-	0.3	-	20	-	69.3	-	100 %
40	-	4510	-	4.4	-	34	-	600	-	0 %
41	-	2812	-	2.8	-	34	-	215.3	-	100 %
42	-	2200	-	3.7	-	32	-	13.8	-	100 %
43	-	4211	-	0.3	-	15	-	43.6	-	100 %
44	-	4841	-	0.3	-	30	-	103.7	-	100 %
45	-	3795	-	0.3	-	18	-	58.1	-	100 %
46	-	5325	-	2.2	-	29	-	600	-	10 %
47	-	5603	-	3.6	-	20	-	468.8	-	70 %
48	-	5863	-	3.1	-	39	-	600	-	0 %
49	-	-	-	-	-	-	-	-	-	-
50	-	-	-	-	-	-	-	-	-	-

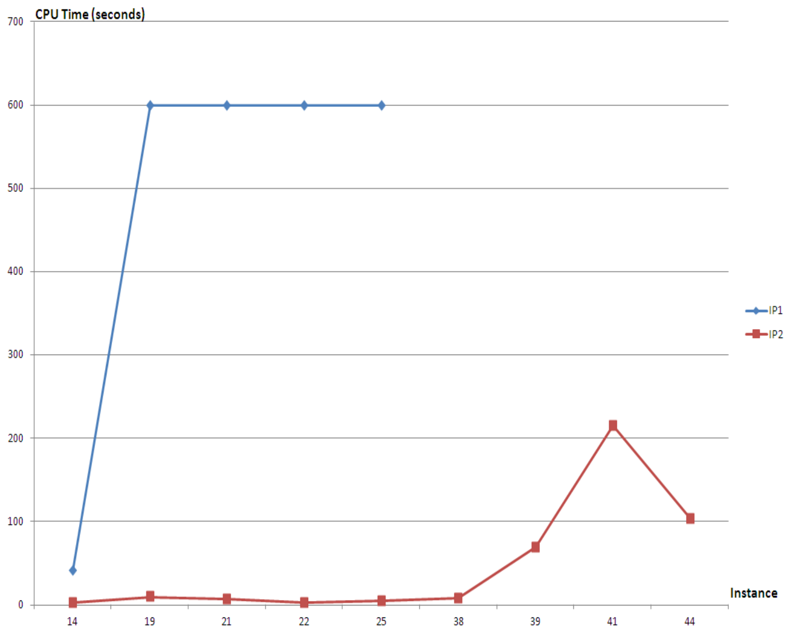


Figure 4.2: Model $IP1$ vs. Model $IP2$, ($\alpha = 1$, $\beta = |N| * T$), CPU times

$(\frac{1}{T-(t-1)} \sum_{\tau=t}^{T} \sum_{f' \in F} \sum_{m' \in M(f')} x_{f',\tau}^{m'} = 0)$, another problem showed up where calculating the losses in qualifications is now dependent on the makespan of the solution. This is in our opinion not realistic, because the calculation of losses in qualifications should be completely independent of time in order to compare the results later with resolution methods. Hence, Model $IP3$ was proposed which is a modified version of Model $IP2$ to take into account both problems.

In this section, the results obtained when solving the instance types using Model $IP3$ are presented and compared to those found by $IP2$. Table 4.6 shows the results when prioritizing the sum of completion times and Table 4.7 when prioritizing the sum of losses in machine qualifications. The results show that the number of losses in machine qualifications for Model $IP3$ is decreased in both tables compared to Model $IP2$. This is true since counting the losses is no longer necessary when there are no jobs to schedule on any of the machines and for all the machines. However, CPU times for Model $IP3$ are greater than those of Model $IP2$ because of the new constraints and variables ($Y_{f,m}$). In addition, whenever the instance is feasible and is solved by the solver with a resolution time less than the time limit, the sum of completion times is found to be the same (Table 4.6). On the other hand, the CPU time for all instance types in Table 4.7 are greater than those in Table 4.6 (Model $IP3$ attains the time limit in more than 50% of the studied instance types in Table 4.7). Instance types 49 and 50 of 80 jobs are still non solvable under the same testing environment conditions. In the rest of this chapter, all the results that are shown for PTC are based on Model $IP3$.

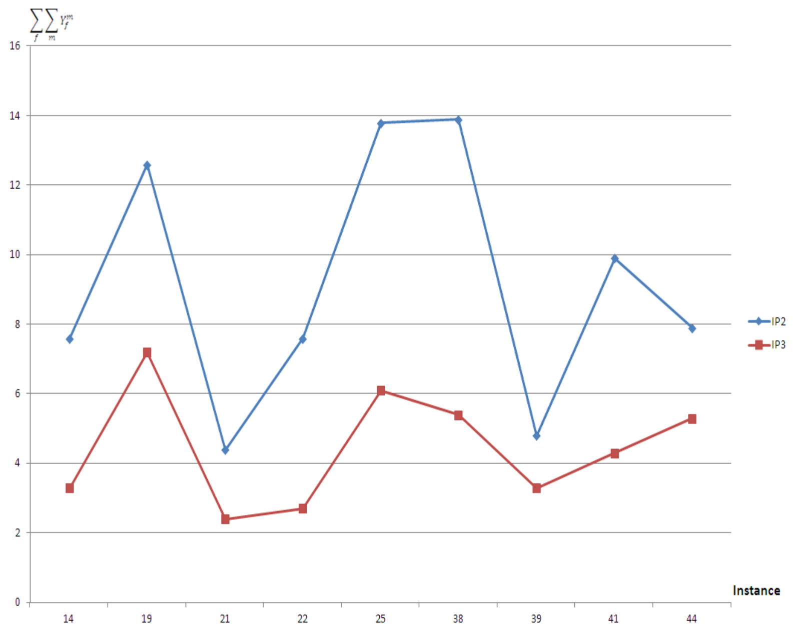


Figure 4.3: Model $IP2$ vs. Model $IP3$, ($\alpha = 1$, $\beta = 1$), $\sum_{f \in F} \sum_{m \in M} Y_f^m$

Figure 4.3 shows the variation of the sum of losses in machine qualifications for Models $IP2$ and $IP3$ when prioritizing the sum of completion times. The difference between both models is noticed where the sum of losses in machine qualifications is now decreased for Model $IP3$. This is a normal behavior since Model $IP2$ is time horizon dependent and hence machine disqualifications are counted even after C_{max}. This is not the case in Model $IP3$ since machine qualifications are now calculated up to every last job on each machine.

Figure 4.4 shows the corresponding variations in CPU times for both models. This figure illustrates again the fact that Model $IP3$ spends more time to solve the selected instance types.

The same observations can be done in Figures 4.5 and 4.6 where machine qualification is prioritized over the sum of completion times. However, regarding the sum of losses in machine qualifications in Figure 4.5, Instance

Table 4.6: Models $IP2$ and $IP3$, ($\alpha = 1$, $\beta = 1$), Minimizing the sum of completion times and the sum of losses in machine qualifications

	$\sum_f C_f$		$\sum_f \sum_m y^m_{f,T}$	$\sum_f \sum_m Y^m_f$	Setups		CPU		% of Opt.	
No.	IP2	IP3	IP2	IP3	IP2	IP3	IP2	IP3	IP2	IP3
1	255	255	0.1	0.0	2	2	0.2	0.4	100 %	100 %
2	245	245	1.4	0.9	3	3	0.3	0.4	100 %	100 %
3	564	564	2.3	1.6	2	2	0.2	0.8	100 %	100 %
4	211	211	4.9	3.1	3	3	0.8	1.7	100 %	100 %
5	278	278	4.8	3.1	3	3	2.9	6.4	100 %	100 %
6	351	351	4.9	1.2	3	3	6.5	16.8	100 %	100 %
7	124	124	4.7	2.1	4	4	1.1	4	100 %	100 %
8	76	76	9.8	3.3	6	6	0.6	2	100 %	100 %
9	116	116	3.9	0.4	5	5	0.1	0.3	100 %	100 %
10	46	46	4.6	0.4	6	6	0	0.1	100 %	100 %
11	96	96	7.7	0.4	6	6	0.1	0.5	100 %	100 %
12	67	67	9.8	0.4	8	8	0.1	0.5	100 %	100 %
13	34	34	9.9	0.3	9	9	0	0.1	100 %	100 %
14	346	346	7.6	3.3	5	5	3.5	13	100 %	100 %
15	285	285	6.7	6.6	8	8	2.7	3.6	100 %	100 %
16	404	404	0.3	0.3	6	6	2.4	3.2	100 %	100 %
17	157	157	5.8	1.2	7	7	1.2	2.2	100 %	100 %
18	399	399	10.8	1.4	7	7	8.4	27.4	100 %	100 %
19	297	297	12.6	7.2	6	6	9.9	47.8	100 %	100 %
20	1163	1163	0.3	0.0	4	4	2.3	4.2	100 %	100 %
21	559	559	4.4	2.4	6	6	2.9	7.7	100 %	100 %
22	483	483	7.6	2.7	6	6	4.4	10.4	100 %	100 %
23	538	538	12.7	7.3	7	11	17.5	95.4	100 %	100 %
24	621	621	6.0	3.4	7	7	12.9	31.2	100 %	100 %
25	427	427	13.8	6.1	11	11	5.1	46.5	100 %	100 %
26	1108	1108	0.3	0.4	4	4	0.5	1.5	100 %	100 %
27	2010	2010	2.1	1.2	4	4	2.8	10.7	100 %	100 %
28	1117	1117	4.1	1.1	5	5	2.5	7.3	100 %	100 %
29	1608	1608	9.8	6.4	3	6	52.6	243.6	100 %	90 %
30	1355	1355	6.8	0.4	8	8	38.2	102.7	100 %	100 %
31	2394	2394	0.4	0.4	3	3	0.5	1.3	100 %	100 %
32	3243	3243	2.3	2.2	3	3	1.5	7.2	100 %	100 %
33	2621	2621	2.4	2.1	6	6	31.1	108.6	100 %	100 %
34	2710	2710	1.8	2.3	5	5	10.6	40.4	100 %	100 %
35	1926	1926	0.3	0.4	7	6	6.9	43.8	100 %	100 %
36	1329	1329	6.8	0.4	6	6	6.6	31.9	100 %	100 %
37	1294	1294	8.8	3.4	8	8	51.5	306.5	100 %	80 %
38	962	962	13.9	5.4	10	10	5.8	78.8	100 %	100 %
39	2992	2992	4.8	3.3	7	7	23.3	137.4	100 %	100 %
40	2504	2504	9.7	9.1	8	8	33.6	345.1	100 %	80 %
41	1158	1158	9.9	4.3	11	11	11.1	106.7	100 %	100 %
42	834	834	8.7	6.2	22	14	6.6	47	100 %	100 %
43	3626	3626	4.9	2.4	4	6	8.7	33.2	100 %	100 %
44	2711	2711	7.9	5.3	9	9	32.1	213.7	100 %	100 %
45	2603	2603	4.8	0.4	6	6	6.8	58.1	100 %	100 %
46	3279	4060	6.8	4.3	8	28	600	600	20 %	10 %
47	3429	3429	8.9	3.1	7	7	68.3	600	100 %	10 %
48	2387	2389	11.8	2.3	12	12	97	600	100 %	0 %
49	-	-	-	-	-	-	-	-	-	-
50	-	-	-	-	-	-	-	-	-	-

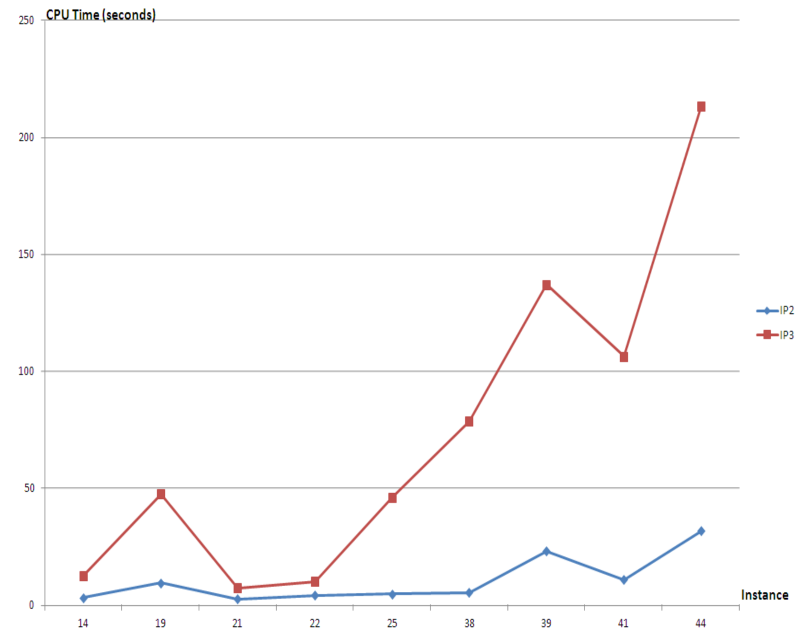


Figure 4.4: Model $IP2$ vs. Model $IP3$, ($\alpha = 1$, $\beta = 1$), CPU times

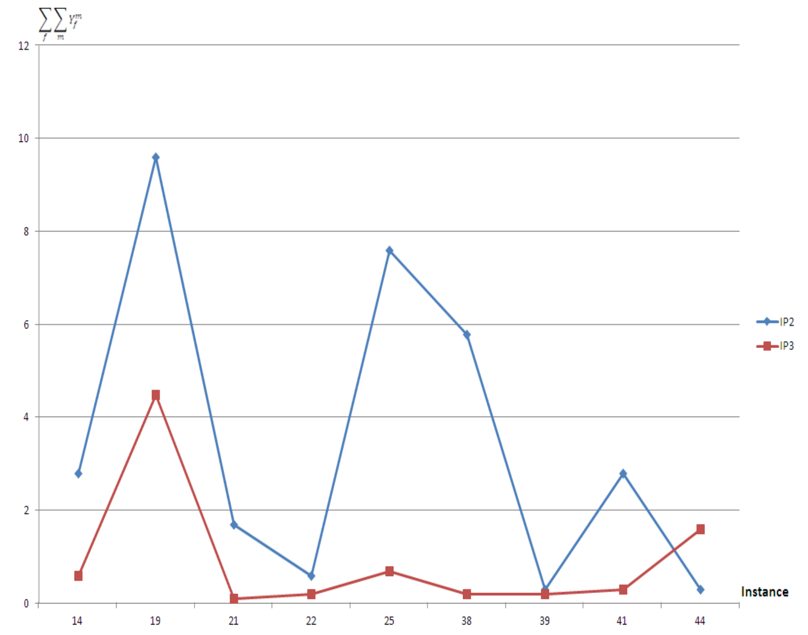


Figure 4.5: Model $IP2$ vs. Model $IP3$, ($\alpha = 1$, $\beta = |N| * T$), $\sum_{f \in F} \sum_{m \in M} Y_f^m$

Table 4.7: Models $IP2$ and $IP3$, ($\alpha = 1$, $\beta = |N| * T$), Minimizing the sum of completion times and the number of losses in machine qualifications

	$\sum_f C_f$		$\sum_f \sum_m y^m_{f,T}$	$\sum_f \sum_m Y^m_f$	Setups		CPU		% of Opt.	
No.	IP2	IP3	IP2	IP3	IP2	IP3	IP2	IP3	IP2	IP3
1	255	255	0.0	0.0	2	2	0.2	0.8	100 %	100 %
2	245	284	0.5	0.0	3	4	0.7	1.8	100 %	100 %
3	810	669	0.5	0.5	5	5	0.2	3.5	100 %	100 %
4	249	231	2.6	1.7	7	6	1.8	17	100 %	100 %
5	298	278	2.7	2.6	5	3	15.1	600	100 %	10 %
6	500	351	1.6	0.6	7	3	5.5	600	100 %	10 %
7	176	133	2.5	0.8	7	6	0.2	2.8	100 %	100 %
8	143	78	6.7	1.9	10	7	0.2	4.9	100 %	100 %
9	184	116	2.7	0.0	7	5	0.4	0.3	100 %	100 %
10	98	46	3.5	0.0	7	6	0	0.1	100 %	100 %
11	174	96	4.6	0.0	6	6	0.5	0.5	100 %	100 %
12	248	67	5.7	0.1	10	8	0.1	0.5	100 %	100 %
13	132	34	7.8	0.0	9	9	0.8	0.1	100 %	100 %
14	782	393	2.8	0.6	17	11	2.9	600	100 %	10 %
15	285	313	6.7	5.8	8	10	3.3	227.5	100 %	100 %
16	404	404	0.2	0.1	6	6	2.4	2.8	100 %	100 %
17	381	163	1.9	0.1	16	8	0.6	1.9	100 %	100 %
18	739	407	4.6	0.1	13	7	5.8	30.1	100 %	100 %
19	441	329	9.6	4.5	9	8	9.7	600	100 %	0 %
20	1163	1163	0.3	0.2	4	4	2.6	3.4	100 %	100 %
21	634	707	1.7	0.1	8	19	7.3	600	100 %	10 %
22	841	512	0.6	0.2	20	10	2.6	230.6	100 %	100 %
23	732	674	8.7	5.6	16	22	600	600	100 %	10 %
24	862	753	2.7	0.9	12	13	13.5	600	100 %	0 %
25	1272	460	7.6	0.7	23	16	5	600	100 %	0 %
26	1108	1108	0.3	0.3	4	4	0.3	1.6	100 %	100 %
27	2614	2064	0.2	0.2	13	6	41.5	29.7	100 %	100 %
28	1370	1135	0.3	0.1	12	6	6.7	18.7	100 %	100 %
29	1972	1734	5.0	3.7	14	14	275.5	600	90 %	10 %
30	1936	1355	0.3	0.1	16	8	41.2	126.8	100 %	100 %
31	2394	2394	0.3	0.1	3	3	0.7	1.2	100 %	100 %
32	3837	3778	0.2	0.2	9	10	103.1	600	100 %	10 %
33	3293	2842	0.2	0.2	20	16	151.5	600	100 %	10 %
34	4033	2966	0.3	0.1	17	10	48.2	600	100 %	0 %
35	1926	1926	0.3	0.1	7	6	6.9	44	100 %	100 %
36	2686	1329	0.9	0.2	23	6	1.5	39.7	100 %	100 %
37	1635	1483	4.8	0.5	16	19	600	600	10 %	10 %
38	2602	1146	5.8	0.2	31	23	8.1	600	100 %	0 %
39	4361	3129	0.3	0.2	20	11	69.3	600	100 %	10 %
40	4510	3421	4.4	7.8	34	37	600	600	0 %	0 %
41	2812	1195	2.8	0.3	34	14	215.3	600	100 %	0 %
42	2200	998	3.7	0.5	32	31	13.8	600	100 %	0 %
43	4211	3820	0.3	0.3	15	11	43.6	600	100 %	0 %
44	4841	3674	0.3	1.6	30	29	103.7	600	100 %	0 %
45	3795	2603	0.3	0.2	18	6	58.1	59.6	100 %	90 %
46	5325	3945	2.2	3.8	29	28	600	600	10 %	0 %
47	5603	4030	3.6	3.9	20	19	468.8	600	70 %	0 %
48	5863	2727	3.1	0.1	39	28	600	600	0 %	0 %
49	-	-	-	-	-	-	-	-	-	-
50	-	-	-	-	-	-	-	-	-	-

type 44 has a value in $IP2$ less than that of $IP3$. This is because the instance type is solved to optimality in Model $IP2$ and not in Model $IP3$.

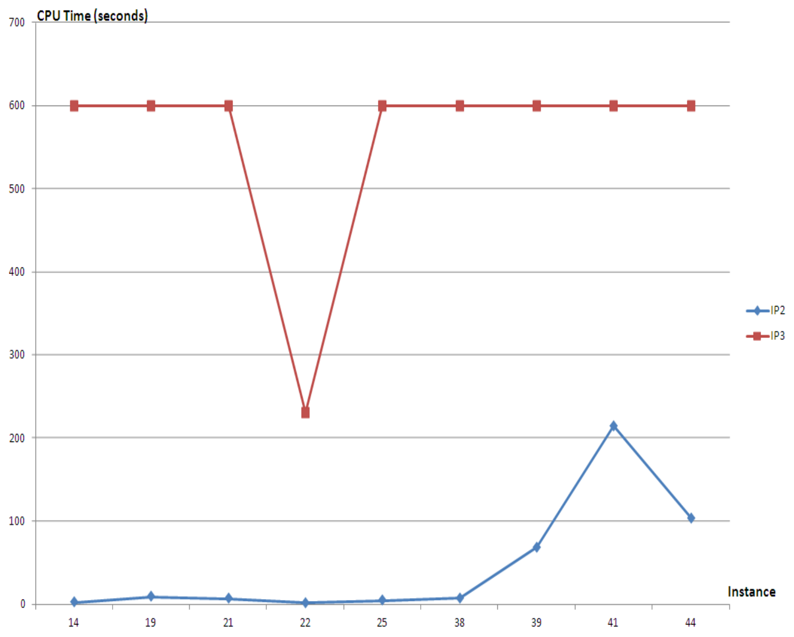


Figure 4.6: Model $IP2$ vs. Model $IP3$, ($\alpha = 1$, $\beta = |N| * T$), CPU times

4.3.2 Model $IP5$ ($PEHF$)

In this section, the results of the tests that were conducted on $PEHF$ using Model $IP5$ are shown. Let us recall that $PEHF$ presents the problem of scheduling job families on *non-identical* parallel machines (different qualification schemes) with setup times and time constraints (γ_f) and with Equipment Health Factor (EHF). We believe that assigning a machine m to process a job family f depends on two major factors: 1 - The criticality of the processed family (Lot/Family Criticality Index, LCI), that may be defined for example as a function of the previous path (machines) of the lot/family as well as other factors including the priority level of the lot/family (hot lot, etc), and 2 - The state of the processing machine which can be described (we assume that it is valid and well defined for the sake of our problem) by its EHF. The consequence of assigning m with a given EHF to process a job family f with a given LCI can be seen as a negative/positive cost that is lost/gained upon this assignment. In $PEHF$, and in Model $IP5$, we consider this cost to be an expected yield chosen between 80% and 100%.

The scheduling decision now should not only take into consideration the loss in machine qualifications, but also the gain in the sum of expected yield resulting from this decision. Let us recall that this yield is define by v_f^m and hence the objective function for Model $IP5$ become a weighted sum of three criteria and is given in Equation 4.1.

$$\max\,(\gamma^{'}\sum_{f\in F}\sum_{m\in M}\sum_{t=1}^{T} w_f x_{f,t}^m v_f^m - (\alpha^{'}\sum_{f\in F} C_f + \beta^{'}\sum_{f\in F}\sum_{m\in M} Y_f^m)) \tag{4.1}$$

Tables 8.a and 8.b summarize the simulation results obtained with Model $IP5$ for different criteria, and when prioritizing each time one or more of the criterion. In what follows, $\alpha^{'} = 1$, $\beta^{'}$ and $\gamma^{'}$ are respectively the weights associated to the sum of completion times, the sum of losses in machine qualifications and the sum of expected yield. We consider that the weights of all families $w_f = 1$, i.e. no family is originally prioritized. Further, $\alpha^{'} = \alpha_o^{'} = 1$, $\beta_o^{'} = |N| * T$ and $\gamma_o^{'} = |M| * |F| * |N| * T$ are the maximum values taken by the weights for prioritizing each criterion. Note that the comparison can be done on the instance types that are feasible and with a CPU time less than the time limit, otherwise, the best solution is kept. In Tables 8.a and 8.b, the following cases are addressed:

- ($\alpha^{'} = \alpha_o^{'}$, $\beta^{'} = 1$, $\gamma^{'} = 1$): Means that the sum of completion times is prioritized compared to the sum of losses in machine qualifications and the sum of the expected yield.
- ($\alpha^{'} = \alpha_o^{'}$, $\beta^{'} = \beta_o^{'}$, $\gamma^{'} = 1$): Means that the sum of losses in machine qualifications is prioritized compared to the sum of completion times and the sum of the expected yield.
- ($\alpha^{'} = \alpha_o^{'}$, $\beta^{'} = 1$, $\gamma^{'} = \gamma_o^{'}$): Means that the sum of the expected yield is prioritized compared to the sum of completion times and the sum of losses in machine qualifications.
- ($\alpha^{'} = \alpha_o^{'}$, $\beta^{'} = \beta_o^{'}$, $\gamma^{'} = \gamma_o^{'}$): Means that the sum of losses in machine qualifications and the sum of the expected yield is prioritized compared to the sum of completion times.

Results for $\sum_{f\in F} C_f$ and $\sum_f \sum_m Y_f^m$ for different ($\alpha^{'}$,$\beta^{'}$,$\gamma^{'}$) combinations (Model $IP5$)

In Table 8.a, the sum of completion times and the corresponding sum of losses in machine qualifications are shown for each combination ($\alpha^{'}$,$\beta^{'}$,$\gamma^{'}$).

In this table, the sum of completion times is optimally minimized for the combination ($\alpha' = \alpha'_o$, $\beta' = 1$, $\gamma' = 1$). This sum is lower than or equal to all other sums, over all the instances, for all other combinations of weights. The second column dominates Columns 3, 4 and 5 regarding minimum value of $\sum_{f \in F} C_f$. However, for the minimum sum of losses in machine qualifications, Column 7 dominates Columns 6, 8 and 9, which corresponds to the values obtained for $\sum_f \sum_m Y_f^m$ for all instances while considering the combination of weights ($\alpha' = \alpha'_o$, $\beta' = \beta'_o$, $\gamma' = 1$). Only in Instance type 47, the value of $\sum_f \sum_m Y_f^m$ ($\alpha' = \alpha'_o$, $\beta' = \beta'_o$, $\gamma' = 1$) does not dominate because this is not a guaranteed optimal solution.

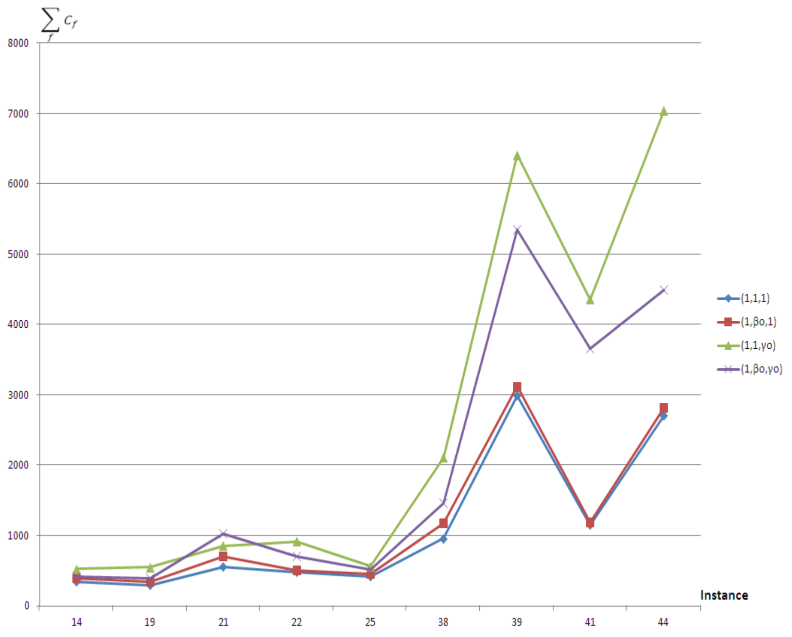


Figure 4.7: Model $IP5$, $\sum_{f \in F} C_f$

The variation of the sum of completion times for different preference vectors for $PEHF$ (Model $IP5$)) is shown in Figure 4.7. Each curve represents the tendency on the selected instances. For a preference vector (1,1,γ_o) where the expected yield is prioritized, the sum of completion times tends to be maximum for all studied preference vectors. However, it tends to be minimum for the preference vector (1,1,1) where the sum of completion times is prioritized over other criteria. In addition, for a preference vector (1,β_o,γ_o) where both machine qualifications and expected yield, the sum of completion times tends to have greater values with the preference vectors (1,β_o,1) and (1,1,1).

Table 8.a: Model $IP5$, ($\alpha^{'} = 1$, $\beta^{'}_o = |N| * T$, $\gamma^{'}_o = |M| * |F| * |N| * T$), $\sum_{f \in F} C_f$ and $\sum_f \sum_m Y^m_f$

	(β,γ) - $\sum_{f \in F} C_f$				(β,γ) - $\sum_f \sum_m Y^m_f$			
$No.$	(1,1)	($\beta^{'}_o$,1)	(1,$\gamma^{'}_o$)	($\beta^{'}_o$,$\gamma^{'}_o$)	(1,1)	($\beta^{'}_o$,1)	(1,$\gamma^{'}_o$)	($\beta^{'}_o$,$\gamma^{'}_o$)
1	255	255	404	404	0.0	0.0	0.1	0.0
2	245	284	303	311	0.9	0.0	0.5	0.0
3	564	669	564	669	1.6	0.9	1.7	1.3
4	211	231	243	231	3.1	1.5	3.6	2.1
5	278	332	338	278	3.1	2.7	3.9	3.4
6	351	351	564	524	1.2	0.6	4.7	2.1
7	124	133	302	133	2.1	0.8	4.8	1.2
8	76	78	164	101	3.3	1.6	9.5	2.4
9	116	116	210	136	0.4	0.3	2.9	0.2
10	46	46	56	50	0.4	0.2	3.6	0.1
11	96	96	195	114	0.4	0.2	4.8	0.1
12	67	67	266	107	0.4	0.1	9.7	0.1
13	34	34	104	50	0.3	0.1	8.8	0.2
14	346	396	527	427	3.3	1.7	6.5	2.4
15	285	313	351	349	6.6	5.8	6.9	6.2
16	404	404	762	762	0.3	0.2	0.4	0.2
17	157	163	203	176	1.2	0.2	2.6	0.1
18	399	407	863	708	1.4	0.2	9.7	1.2
19	297	348	552	400	7.2	4.5	10.7	5.3
20	1163	1163	1687	1687	0.0	0.1	0.4	0.1
21	559	707	856	1033	2.4	0.2	2.9	1.1
22	483	512	918	711	2.7	0.3	5.8	0.1
23	538	574	1141	1046	7.3	6.6	12.6	11.8
24	621	666	1440	1439	3.4	1.9	6.9	3.2
25	427	460	566	520	6.1	0.8	9.5	1.4
26	1108	1108	4585	2550	0.4	0.2	0.3	0.2
27	2010	2064	3324	3113	1.2	0.2	1.7	1.3
28	1117	1135	2161	1908	1.1	0.1	4.6	0.2
29	1608	1776	2691	1960	6.4	4.6	8.6	6.2
30	1355	1355	2855	1981	0.4	0.1	3.8	0.1
31	2394	2394	2925	2925	0.4	0.1	0.3	0.1
32	3243	3765	3457	3763	2.2	0.2	1.7	0.2
33	2621	2752	4963	4708	2.1	0.2	2.5	3.4
34	2710	2977	6133	6881	2.3	0.1	1.8	2.3
35	1926	1926	4040	4040	0.4	0.1	0.3	0.1
36	1329	1329	3948	2659	0.4	0.2	6.7	0.2
37	1294	1403	4116	3636	3.4	1.8	9.9	7.1
38	962	1183	2116	1466	5.4	0.2	10.6	1.3
39	2992	3125	6419	5357	3.3	0.3	4.8	0.3
40	2504	3281	5928	3876	9.1	7.5	9.7	9.8
41	1158	1189	4366	3670	4.3	0.3	9.9	7.2
42	834	1020	1944	2980	6.2	0.6	7.9	5.4
43	3626	3825	4768	5076	2.4	0.3	4.8	0.3
44	2711	2820	7047	4497	5.3	0.9	8.6	4.2
45	2603	2603	8248	6355	0.4	0.3	4.5	2.3
46	3985	4118	6340	4966	4.3	3.7	4.7	4.1
47	3503	4214	8460	7247	3.1	5.8	6.7	7.4
48	2387	2531	5046	3763	2.3	1.6	9.6	2.1
49	-	-	-	-	-	-	-	-
50	-	-	-	-	-	-	-	-

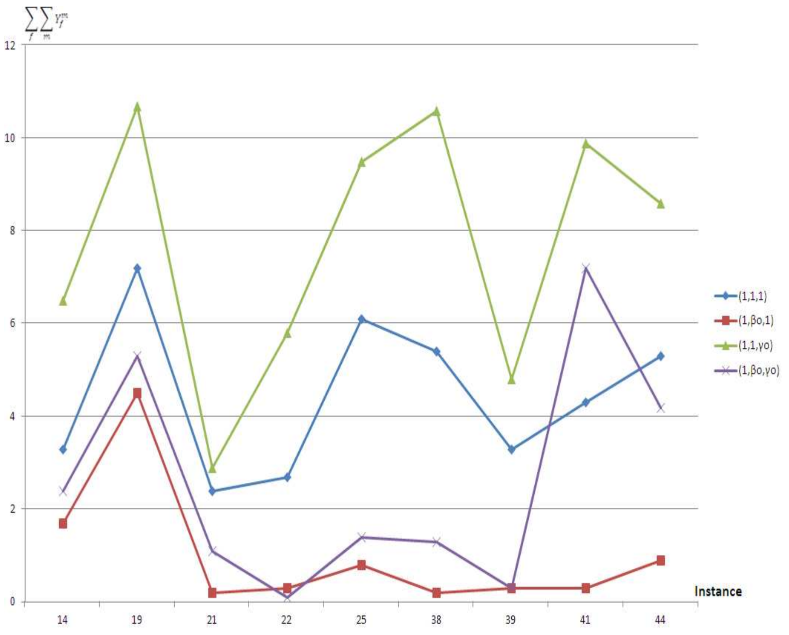


Figure 4.8: Model $IP5$, $\sum_f \sum_m Y_f^m$

The sum of losses in machine qualifications is minimized for the preference vector (1,β_o,1) as Figure 4.8 illustrates. This is because machine qualifications are prioritized. Also, note that, for the preference vector (1,β_o,γ_o) where machine qualifications and expected yield are both prioritized, the sum of losses in machine qualifications also tends to be lower than with preference vectors (1,1,1) and (1,1,γ_o). However, in Instance type 41 for example, the sum of losses in machine qualifications is greater than with the preference vector (1,1,1) because the latter does not guarantee optimality as shown in Figure 4.10.

Results for yield and CPU time for different (α',β',γ') combinations (Model $IP5$)

In Table 8.b, the sum of expected yield and the corresponding CPU time are shown for each of the (α',β',γ') combinations. The sum of expected yield is optimally maximized for the combination of ($\alpha' = \alpha_o'$, $\beta' = 1$, $\gamma' = \gamma_o'$). This sum is greater than or equal to all other sums, over all the instances, for all other different combinations of weights. The fourth column dominates Columns 2, 3 and 5 on the maximum value of $\sum_{t=1}^{T} \sum_f \sum_m x_{f,t}^m v_{f,t}^m$.

The CPU time either when prioritizing the sum of expected yield or the sum of completion times is lower than when prioritizing the sum of losses in machine qualification and the sum of expected yield, i.e. with the combinations of weights ($\alpha^{'} = \alpha^{'}_{o}$, $\beta^{'} = \beta^{'}_{o}$, $\gamma^{'} = 1$) and ($\alpha^{'} = \alpha^{'}_{o}$, $\beta^{'} = \beta^{'}_{o}$, $\gamma^{'} = \gamma^{'}_{o}$). This also illustrates the added complexity when the notion of machine qualification and time constraint is considered.

Table 8.b: Model $IP5$, ($\alpha^{'} = 1$, $\beta^{'}_{o} = |N| * T$, $\gamma^{'}_{o} = |M| * |F| * |N| * T$), $\sum_{t=1}^{T} \sum_{f} \sum_{m} x^{m}_{f,t} v^{m}_{f,t}$ and CPU Time

	(β,γ) - $\sum_{t=1}^{T} \sum_{f} \sum_{m} x^{m}_{f,t} v^{m}_{f,t}$				(β,γ) - CPU Time				(β,γ) - % of Opt.			
$No.$	(1,1)	($\beta^{'}_{o}$,1)	(1,$\gamma^{'}_{o}$)	($\beta^{'}_{o}$,$\gamma^{'}_{o}$)	(1,1)	($\beta^{'}_{o}$,1)	(1,$\gamma^{'}_{o}$)	($\beta^{'}_{o}$,$\gamma^{'}_{o}$)	(1,1)	($\beta^{'}_{o}$,1)	(1,$\gamma^{'}_{o}$)	($\beta^{'}_{o}$,$\gamma^{'}_{o}$)
1	870	870	922	922	0.8	0.8	2.8	2.1	100 %	100 %	100 %	100 %
2	890	888	894	892	0.4	1.8	0.2	2	100 %	100 %	100 %	100 %
3	952	952	952	952	0.8	4	0.8	3.3	100 %	100 %	100 %	100 %
4	848	844	856	844	2.3	15.6	2.5	21.8	100 %	100 %	100 %	100 %
5	854	835	856	854	6.4	600	22.9	600	100 %	10 %	100 %	10 %
6	879	879	896	883	16.8	600	600	600	100 %	10 %	20 %	10 %
7	905	909	939	909	3	3.8	5.7	27.6	100 %	100 %	100 %	100 %
8	940	939	981	954	3	9.7	3.1	8.3	100 %	100 %	100 %	100 %
9	868	868	915	890	0.3	0.3	0.3	1.2	100 %	100 %	100 %	100 %
10	911	911	926	924	0.1	0.1	0.1	0.1	100 %	100 %	100 %	100 %
11	905	905	938	918	0.5	0.5	0.6	1	100 %	100 %	100 %	100 %
12	905	905	982	948	0.8	0.7	4.4	7.8	100 %	100 %	100 %	100 %
13	908	908	950	939	0.2	0.2	0.2	0.2	100 %	100 %	100 %	100 %
14	1826	1837	1868	1853	12.4	600	4.2	600	100 %	10 %	100 %	10 %
15	1813	1818	1835	1835	5.1	600	1.2	245	100 %	10 %	100 %	100 %
16	1794	1794	1892	1892	5.5	3.7	1.1	1.1	100 %	100 %	100 %	100 %
17	1822	1815	1837	1820	1.2	3.5	0.3	4.6	100 %	100 %	100 %	100 %
18	1730	1738	1844	1787	26.9	30.8	39.5	600	100 %	100 %	100 %	10 %
19	1749	1755	1839	1765	58.8	600	21.6	600	100 %	10 %	100 %	10 %
20	2698	2698	2748	2748	3.4	3.1	1.3	1.3	100 %	100 %	100 %	100 %
21	2732	2664	2840	2833	7.6	600	1.9	299.2	100 %	0 %	100 %	100 %
22	2663	2693	2788	2747	14.3	310.6	6.5	600	100 %	90 %	100 %	20 %
23	2704	2679	2785	2761	128.1	600	69.3	600	100 %	0 %	100 %	20 %
24	2770	2758	2900	2875	30.6	600	24.4	600	100 %	0 %	100 %	10 %
25	2634	2618	2664	2644	38.7	600	15.3	600	100 %	10 %	100 %	10 %
26	3746	3746	3941	3936	1.6	1.4	12.9	9.5	100 %	100 %	100 %	100 %
27	3595	3606	3814	3814	11.8	30.9	9.4	57.5	100 %	100 %	100 %	100 %
28	3538	3534	3582	3568	7.8	21	18.7	548.1	100 %	100 %	100 %	60 %
29	3616	3584	3680	3583	241.5	600	600	600	90 %	0 %	20 %	10 %
30	3568	3568	3686	3666	106.3	115.2	52	414.1	100 %	100 %	100 %	70 %
31	4691	4691	4700	4700	1.2	1.2	0.8	0.8	100 %	100 %	100 %	100 %
32	4210	4194	4210	4194	8.2	600	5.4	421.8	100 %	0 %	100 %	80 %
33	4393	4378	4455	4454	78.2	600	600	600	100 %	0 %	10 %	0 %
34	4602	4597	4744	4708	45.9	600	600	600	100 %	0 %	10 %	0 %
35	4346	4346	4608	4608	43.9	44	33.1	33	100 %	100 %	100 %	10 %
36	4544	4544	4824	4776	31.1	35	23.8	600	100 %	100 %	100 %	0 %
37	4328	4356	4764	4636	300.4	600	97	600	80 %	0 %	100 %	0 %
38	4490	4470	4615	4561	82.5	600	36.2	600	100 %	0 %	100 %	0 %
39	5347	5354	5585	5528	148.1	600	96.2	600	100 %	10 %	100 %	0 %
40	5449	5383	5747	5626	426.6	600	347.6	600	80 %	0 %	50 %	0 %
41	5406	5324	5799	5787	118.9	600	51.6	600	100 %	0 %	100 %	0 %
42	5328	5303	5550	5519	48.6	600	37.6	600	100 %	0 %	100 %	0 %
43	6275	6266	6446	6435	34.6	600	33.9	600	100 %	0 %	100 %	0 %
44	6139	6099	6427	6242	228.2	600	600	600	100 %	0 %	0 %	0 %
45	6396	6396	6757	6741	60	58.9	91.5	600	100 %	0 %	100 %	0 %
46	6286	6266	6560	6370	600	600	600	600	10 %	10 %	0 %	0 %
47	6190	6200	6625	6418	600	600	425.5	600	10 %	0 %	50 %	0 %
48	6259	6231	6426	6195	600	600	293.2	600	0 %	0 %	90 %	0 %
49	-	-	-	-	-	-	-	-	-	-	-	-
50	-	-	-	-	-	-	-	-	-	-	-	-

The percentage of gain on the expected yield obtained with the preference

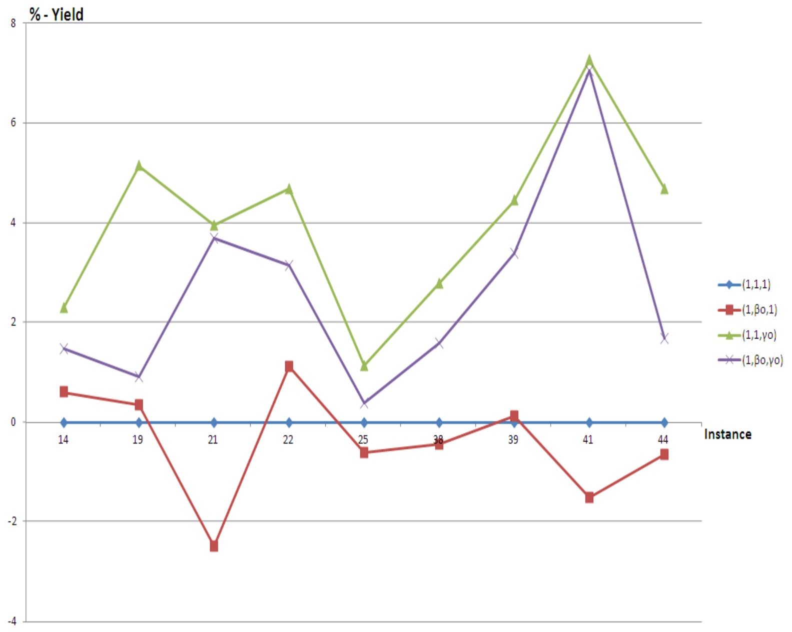


Figure 4.9: Model $IP5$, Percentage of yield gain

vector (1,1,1), where the sum of completion times is prioritized compared to other criteria is shown in Figure 4.9. This figure shows that the gain is maximized for the preference vector (1,1,γ_o) where the expected yield is prioritized. However, this gain tends to be minimal for the preference vector (1,β_o,1) where machine qualification is prioritized compared to other criteria. In addition, for the preference vector (1,β_o,γ_o), this percentage is larger than with preference vectors (1,1,1) and (1,β_o,1), but smaller than with preference vector (1,1,γ_o). Further, Figure 4.10 represents the CPU times for all studied preference vectors. It shows that whenever machine qualification is prioritized with and compared to the expected yield and sum of completion times, CPU times are the largest and close or equal to the time limit.

4.4 Problem analysis

4.4.1 Analysis of the objective function (PTC and $PEHF$)

Problem with Time Constraint (PTC)

In Tables 9.a and 9.b, we show the results when both criteria of the objective function in PTC are optimized, using the following values: $\alpha = 1$ and

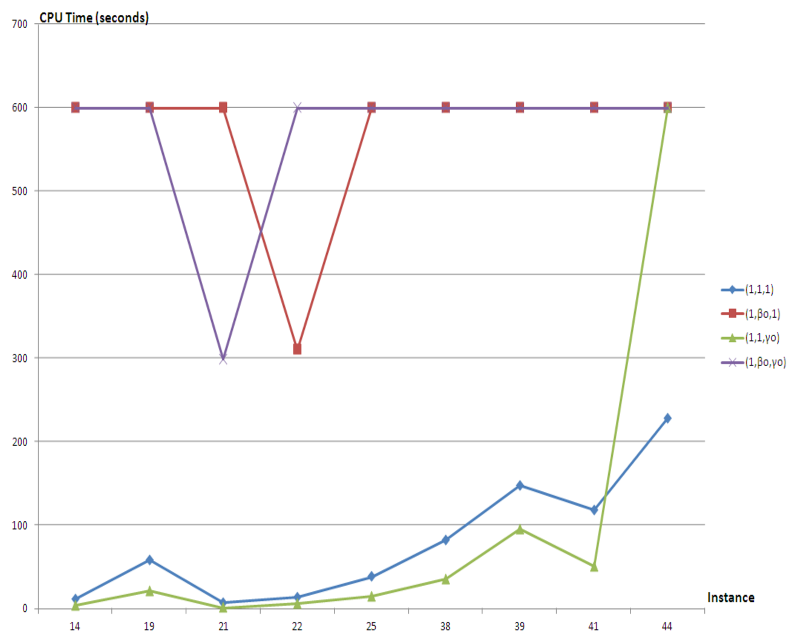


Figure 4.10: Model $IP5$, CPU times

$\beta = k * \beta_o$, where $\beta_o = |N| * T$ and $k = 1/\beta_o$, 1/6, 1/4, 1/2 and 1. These values are chosen so that the scheduling criterion, minimizing the sum of completion times, is second to the minimization of the number of disqualifications except for $k = 1/\beta_o$, so that it is equivalent to a lexicographical order. The weight of the qualification criterion is increased as a factor of fractions of β_o aiming at investigating the effect of its variation on the values of the objective function. The results show that the problem becomes much harder directly after increasing the weight of the qualification criterion, even only with $k = 1/6$. This is due to the fact that the goal is not only to find a feasible schedule maximizing the satisfaction of the threshold constraints, but also to balance with the minimization of the sum of completion times. On the other hand, for the instance types with a resolution time below the time limit, the values of $\sum_f \sum_m Y_f^m$ are smaller as the value of the weight of this criterion is increased. However, the values of $\sum_f C_f$ increase. This is an expected behavior and the antagonistic nature of both criteria is evidenced.

Figure 4.11 shows the percentage increase of the sum of completion times when varying the weight of machine qualifications (β). As the figure shows, the percentage of increase tends to be smaller for small values of β. These values of percentages are based on the values obtained for the preference vector (1,1), where the sum of completion times is prioritized. Note that it is not guaranteed for higher values of β, that the percentage increase of the sum of

Table 9.a: Model $IP3$, ($\alpha = 1$, $\beta_o = |N| * T$), $\sum_f C_f$ and $\sum_f \sum_m Y_f^m$ with β variation

$No.$	β - $\sum_f C_f$					β - $\sum_f \sum_m Y_f^m$				
	1	$\beta_o/6$	$\beta_o/4$	$\beta_o/2$	β_o	1	$\beta_o/6$	$\beta_o/4$	$\beta_o/2$	β_o
1	255	255	255	255	255	0.0	0.0	0.0	0.0	0.0
2	245	284	284	284	284	0.9	0.4	0.3	0.3	0.0
3	564	669	669	669	669	1.6	1.3	1.2	0.9	0.5
4	211	231	231	231	231	3.1	1.9	1.8	1.8	1.7
5	278	326	369	278	278	3.1	2.1	1.9	2.8	2.6
6	351	351	351	351	351	1.2	0.9	0.9	0.8	0.6
7	124	133	133	133	133	2.1	1.2	1.1	0.9	0.8
8	76	78	78	78	78	3.3	2.4	2.2	2.2	1.9
9	116	116	116	116	116	0.4	0.0	0.0	0.0	0.0
10	46	46	46	46	46	0.4	0.1	0.0	0.0	0.0
11	96	96	96	96	96	0.4	0.1	0.1	0.2	0.0
12	67	67	67	67	67	0.4	0.2	0.2	0.4	0.1
13	34	34	34	34	34	0.3	0.2	0.2	0.1	0.0
14	346	393	393	393	393	3.3	1.4	1.2	1.1	0.6
15	285	313	313	313	313	6.6	6.3	6.3	6.2	5.8
16	404	404	404	404	404	0.3	0.2	0.2	0.1	0.1
17	157	163	163	163	163	1.2	0.4	0.4	0.3	0.1
18	399	407	407	407	407	1.4	0.4	0.3	0.3	0.1
19	297	368	356	363	329	7.2	4.4	4.1	4.1	4.5
20	1163	1163	1163	1163	1163	0.0	0.2	0.2	0.1	0.2
21	559	711	707	707	707	2.4	0.2	0.1	0.1	0.1
22	483	512	512	512	512	2.7	0.2	0.1	0.1	0.2
23	538	738	738	872	674	7.3	6.6	6.6	6.4	5.6
24	621	694	669	689	753	3.4	2.4	2.3	2.2	0.9
25	427	460	460	466	460	6.1	1.4	1.1	1.1	0.7
26	1108	1108	1108	1108	1108	0.4	0.2	0.1	0.1	0.3
27	2010	2064	2064	2064	2064	1.2	0.2	0.2	0.1	0.2
28	1117	1135	1135	1135	1135	1.1	0.2	0.1	0.1	0.1
29	1608	1791	1850	1761	1734	6.4	4.4	4.4	4.9	3.7
30	1355	1355	1355	1355	1355	0.4	0.1	0.1	0.1	0.1
31	2394	2394	2394	2394	2394	0.4	0.2	0.2	0.1	0.1
32	3243	3879	3890	3878	3778	2.2	0.2	0.2	0.1	0.2
33	2621	2812	2792	2682	2842	2.1	0.4	0.7	0.6	0.2
34	2710	2981	2970	2966	2966	2.3	0.4	0.2	0.1	0.1
35	1926	1926	1926	1926	1926	0.4	0.4	0.3	0.3	0.1
36	1329	1329	1329	1329	1329	0.4	0.3	0.2	0.2	0.2
37	1294	1504	1412	1618	1483	3.4	2.3	2.3	1.1	0.5
38	962	1156	1181	1137	1146	5.4	0.4	0.4	0.4	0.2
39	2992	3125	3135	3125	3129	3.3	0.4	0.3	0.3	0.2
40	2504	3421	3421	3421	3421	9.1	8.2	8.2	7.8	7.8
41	1158	1190	1197	1193	1195	4.3	0.4	0.4	0.3	0.3
42	834	1009	1011	1109	998	6.2	1.2	1.2	1.1	0.5
43	3626	3825	3822	3820	3820	2.4	0.4	0.3	0.3	0.3
44	2711	2830	2842	3306	3674	5.3	0.4	2.9	2.7	1.6
45	2603	2603	2603	2603	2603	0.4	0.4	0.3	0.4	0.2
46	4060	3947	4037	3989	3945	4.3	4.1	4.1	4.3	3.8
47	3429	4159	4151	4181	4030	3.1	5.8	5.9	5.7	3.9
48	2389	2575	2503	2581	2727	2.3	1.7	1.6	1.6	0.1
49	-	-	-	-	-	-	-	-	-	-
50	-	-	-	-	-	-	-	-	-	-

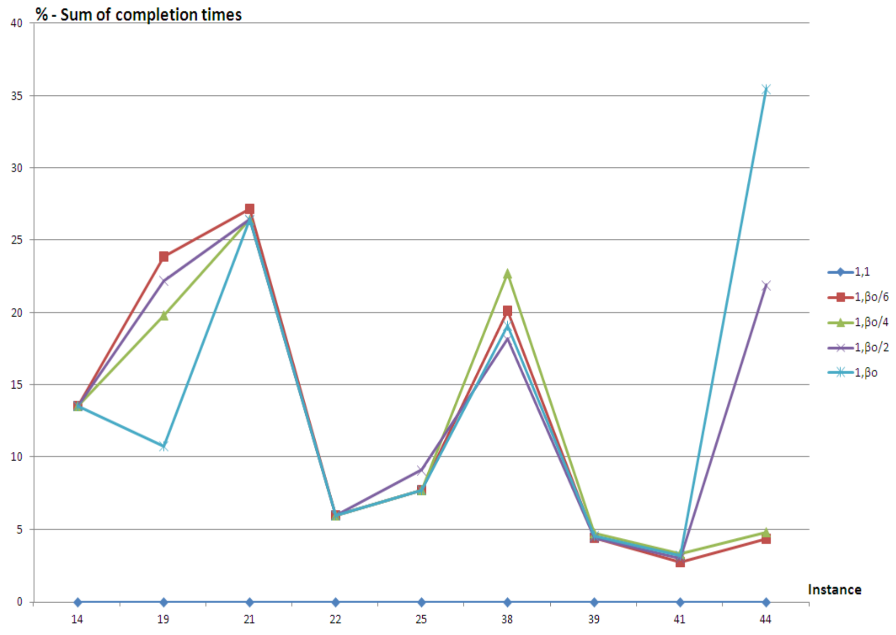


Figure 4.11: Model $IP3$, percentage increase of $\sum_f C_f$, variations of β

completion times is directly proportional to this variation when there is no complete dominance of one criterion over the other, i.e. preference vectors (1,$\beta_o/2$), (1,$\beta_o/4$) and (1,$\beta_o/6$). Instance type 21 is an example.

Figure 4.12 shows the variation of the sum of losses in machine qualifications for the selected instances and for different preference vectors. The sum of losses in machine qualifications is maximal for the preference vector (1,1) where the sum of completion times is prioritized, and tends to be minimal for the preference vector (1,β_o), where machine qualification is prioritized. This illustrates again the antagonistic nature of both criteria. It is important to mention that the solution times for Instance types 19 and 44 reach the time limit, and hence their behavior is not contradictory with the antagonistic nature of both criteria since only the best found solution is shown.

On the contrary, the number of setups is minimized as the sum of completion time is prioritized. This can be seen in Figure 4.13 where the preference vector (1,1) leads to a minimum number of setups for all the selected instance types.

Further, note that in Figure 4.14, CPU times increase whenever the criterion on machine qualifications becomes important, i.e. preference vectors

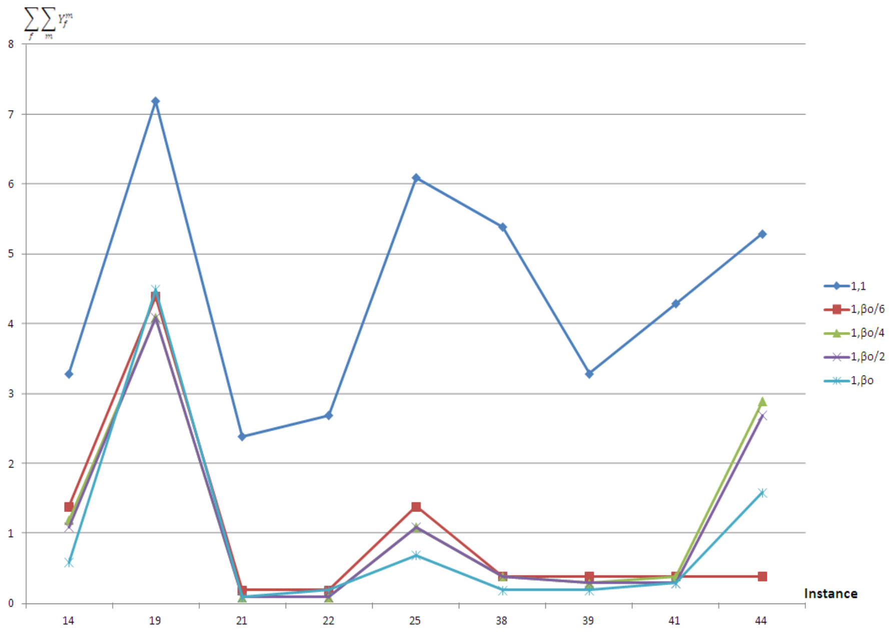


Figure 4.12: Model $IP3$, $\sum_f \sum_m Y_f^m$, variations of β

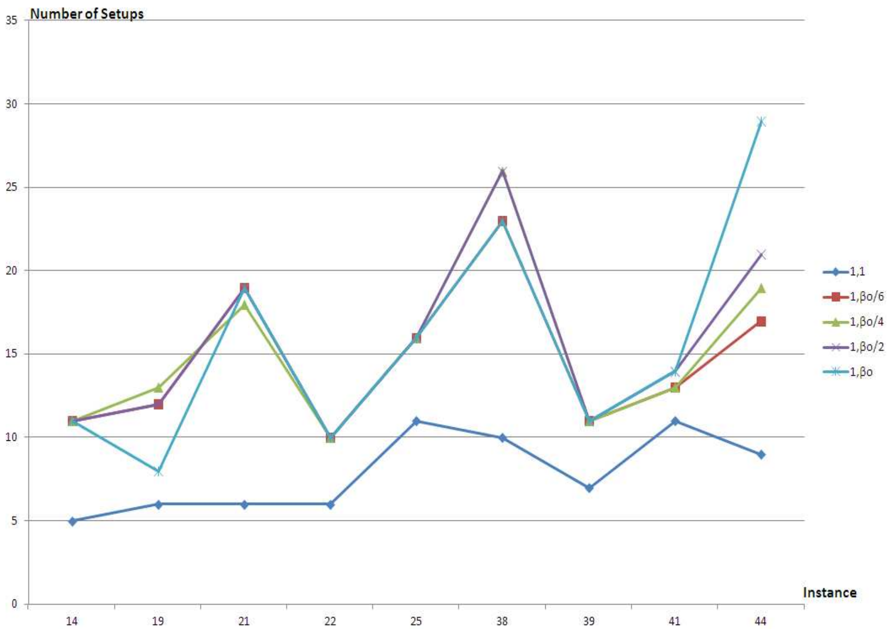


Figure 4.13: $IP3$ - Number of setups - β variation

(1,β_o/6), (1,β_o/4), (1,β_o/2) and (1,β_o). This is another evidence of the increase of the problem complexity when time constraints are included.

Table 9.b: Model $IP3$, ($\alpha = 1$, $\beta_o = |N| * T$), Number of needed setups and CPU time with variations of β

	β - Setups					β - (CPU)					β - % of Opt.				
No.	1	$\beta_o/6$	$\beta_o/4$	$\beta_o/2$	β_o	1	$\beta_o/6$	$\beta_o/4$	$\beta_o/2$	β_o	1	$\beta_o/6$	$\beta_o/4$	$\beta_o/2$	β_o
1	2	2	2	2	2	0.3	0.8	0.7	0.7	0.8	100 %	100 %	100 %	100 %	100 %
2	3	4	4	4	4	0.4	1.7	1.8	1.7	1.8	100 %	100 %	100 %	100 %	100 %
3	2	5	5	5	5	0.8	12.5	11.9	8.1	11.9	100 %	100 %	100 %	100 %	100 %
4	3	7	7	7	6	1.7	16.4	19.1	23.7	19.1	100 %	100 %	100 %	100 %	100 %
5	3	3	5	3	3	6.4	600	600	600	600	100 %	10 %	10 %	10 %	10 %
6	3	3	3	3	3	16.8	600	600	600	600	100 %	10 %	10 %	10 %	10 %
7	4	6	6	6	6	4	2.8	2.9	3.4	2.9	100 %	100 %	100 %	100 %	100 %
8	6	7	7	7	7	2	5	5.5	6.2	5.5	100 %	100 %	100 %	100 %	100 %
9	5	5	5	5	5	0.3	0.3	0.3	0.3	0.3	100 %	100 %	100 %	100 %	100 %
10	6	6	6	6	6	0.1	0.1	0.1	0.1	0.1	100 %	100 %	100 %	100 %	100 %
11	6	6	6	6	6	0.5	0.5	0.5	0.5	0.5	100 %	100 %	100 %	100 %	100 %
12	8	8	8	8	8	0.5	0.5	0.5	0.6	0.5	100 %	100 %	100 %	100 %	100 %
13	9	9	9	9	9	0.1	0.2	0.2	0.2	0.2	100 %	100 %	100 %	100 %	100 %
14	5	11	11	11	11	13	335.6	206.1	600	206.1	100 %	70 %	100 %	10 %	10 %
15	8	12	11	12	10	3.6	107.4	600	387.9	600	100 %	100 %	10 %	70 %	100 %
16	6	6	6	6	6	3.2	3.6	3.5	3.2	3.5	100 %	100 %	100 %	100 %	100 %
17	7	8	8	8	8	2.2	1.6	2.1	2.2	2.1	100 %	100 %	100 %	100 %	100 %
18	7	7	7	7	7	27.4	28.6	30.7	30.5	30.7	100 %	100 %	100 %	100 %	100 %
19	6	12	13	12	8	47.8	600	600	600	600	100 %	10 %	10 %	10 %	0 %
20	4	4	4	4	4	4.2	3.6	3.5	3.7	3.5	100 %	100 %	100 %	100 %	100 %
21	6	19	18	19	19	7.7	362.3	373.7	509.6	373.7	100 %	60 %	60 %	30 %	10 %
22	6	10	10	10	10	10.4	171.5	155.7	182.3	155.7	100 %	100 %	100 %	100 %	100 %
23	11	21	21	26	22	95.4	600	600	600	600	100 %	0 %	0 %	0 %	10 %
24	7	12	11	10	13	31.2	600	600	600	600	100 %	0 %	0 %	0 %	0 %
25	11	16	16	16	16	46.5	600	600	600	600	100 %	0 %	0 %	0 %	0 %
26	4	4	4	4	4	1.5	1.5	1.6	1.6	1.6	100 %	100 %	100 %	100 %	100 %
27	4	6	6	6	6	10.7	30.5	29.3	29.9	29.3	100 %	100 %	100 %	100 %	100 %
28	5	6	6	6	6	7.3	19.3	19.1	19.1	19.1	100 %	100 %	100 %	100 %	100 %
29	6	15	18	16	14	243.6	600	600	600	600	90 %	0 %	0 %	0 %	10 %
30	8	8	8	8	8	102.7	129.1	129.4	129.6	129.4	100 %	100 %	100 %	100 %	100 %
31	3	3	3	3	3	1.3	1.2	1.3	1.2	1.3	100 %	100 %	100 %	100 %	100 %
32	3	10	11	11	10	7.2	600	600	600	600	100 %	0 %	0 %	0 %	10 %
33	6	15	15	10	16	108.6	600	600	600	600	100 %	0 %	0 %	0 %	10 %
34	5	12	12	10	10	40.4	600	600	600	600	100 %	0 %	0 %	0 %	0 %
35	6	7	6	6	6	43.8	44.8	43.9	45.4	43.9	100 %	100 %	100 %	100 %	100 %
36	6	6	6	6	6	31.9	38.5	43.9	39.5	43.9	100 %	100 %	100 %	100 %	100 %
37	8	20	19	22	19	306.5	600	600	600	600	80 %	0 %	0 %	0 %	10 %
38	10	23	26	26	23	78.8	600	600	600	600	100 %	0 %	0 %	0 %	0 %
39	7	11	11	11	11	137.4	600	600	600	600	100 %	0 %	0 %	0 %	10 %
40	8	37	37	37	37	345.1	600	600	600	600	80 %	0 %	0 %	0 %	0 %
41	11	13	13	14	14	106.7	600	600	600	600	100 %	0 %	0 %	0 %	0 %
42	14	25	24	33	31	47	600	600	600	600	100 %	0 %	0 %	0 %	0 %
43	6	13	12	11	11	33.2	600	600	600	600	100 %	0 %	0 %	0 %	0 %
44	9	17	19	21	29	213.7	600	600	600	600	100 %	0 %	0 %	0 %	0 %
45	6	6	6	6	6	58.1	60	59.9	60.1	59.9	100 %	90 %	90 %	90 %	90 %
46	28	28	30	32	28	600	600	600	600	600	10 %	0 %	0 %	0 %	0 %
47	7	25	25	21	19	600	600	600	600	600	10 %	0 %	0 %	0 %	0 %
48	12	18	20	19	28	600	600	600	600	600	0 %	0 %	0 %	0 %	0 %
49	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
50	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Problem with Equipment Health Factor ($PEHF$)

In Tables 11.e and 10.b, we show the results when the three criteria the objective function of $PEHF$ are optimized, using the following values: $\alpha^{'} = 1$, $\beta^{'} = \beta_o = |N| * T$ and $\gamma^{'} = k * \gamma_o$, where $\gamma_o = |M| * |F| * |N| * T$ and $k = 1/\gamma_o$, 1/8, 1/6, 1/4, 1/2 and 1. These values are chosen so that the scheduling criterion, minimizing the sum of completion times, is second to

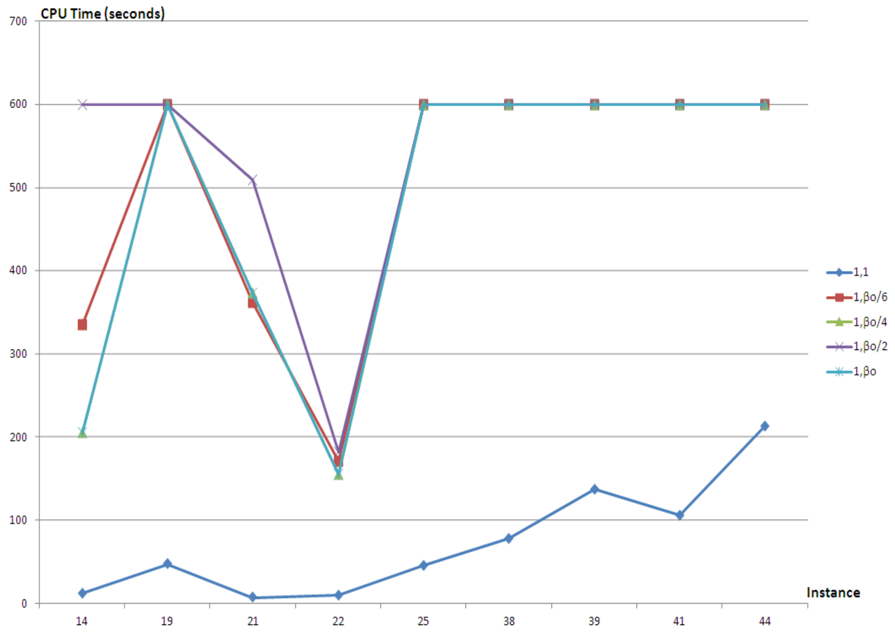


Figure 4.14: $IP3$ - CPU times - β variation

both the minimization of the number of disqualifications and the sum of expected yield. The weight of the yield criterion is increased as a factor of fractions of γ_o, to investigate the effect of $\gamma^{'}$ on the objective function. The tests are conducted while varying the coefficient multiplying the yield criterion by prioritizing in all cases the sum of losses in machine qualifications. This criterion is prioritized because $PEHF$ is considered as an extension of PTC, and hence minimizing the disqualifications should always be taken into account. The results show that, for larger values of k, the yield is better maximized for all instance types for which CPU time is below the time limit. The sum of losses in machines qualifications is simultaneously minimized and has smaller values for smaller values of k. However, the value of $\sum_f C_f$ are increasing.

The values of α' and β' are respectively fixed to 1 and β_o when studying the effect of varying γ on the different criteria of the objective function. The value of β is fixed to β_o because we consider that $PEHF$ is an extension of PTC and that machine qualifications are important. Also, β_o is used to test the antagonistic nature of machine qualifications and expected yield. In Figure 4.15, the percentage of gain on the expected yield for the selected instance types is shown based on the values obtained for the preference vector (1,β_o,1), where machine qualifications are prioritized. As the figure shows, the percentage obtained for the preference vector (1,β_o,γ_o), i.e. prioritizing machine qualifications and expected yield, is maximized compared to other preference vectors where the priority of the expected yield is gradually decreased ($\gamma = \gamma_o/2$, $\gamma_o/4$, $\gamma_o/6$ and $\gamma_o/8$). Once the weight of the expected yield is decreased and consequently the weight of machine qualifications is relatively increased, the percentage of gain on the expected yield decreases and the number of machine qualifications decreases as well. This is shown in Figure 4.16 where the values of losses in machine qualifications are provided for all preference vectors. This is true for instance types that are solved to optimality and the tendency in Figures 4.15 and 4.16 shows this hypothesis. Therefore, the antagonistic nature of machine qualifications and expected yield is illustrated, in addition to the competing criteria of machine qualifications and the sum of completion times discussed in previous sections.

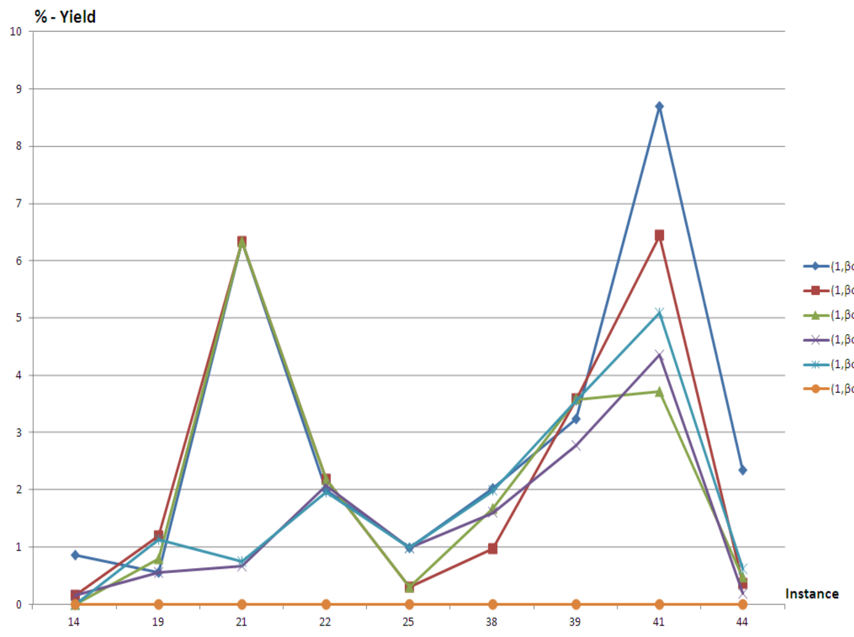


Figure 4.15: Model $IP5$, gain in percentage on yield for different values γ, based on the yield obtained for $\gamma = 1$

Furthermore, the effect of varying the weight of the expected yield on the sum of completion times is shown in Figure 4.17. The tendency is to lose on the sum of completion times (increase) when the priority on the expected yield is increased. This illustrates the antagonistic nature between the sum of completion times and the expected yield. However, this is not always true since, if Instance type 27 in Table 10.b is considered for which the optimal

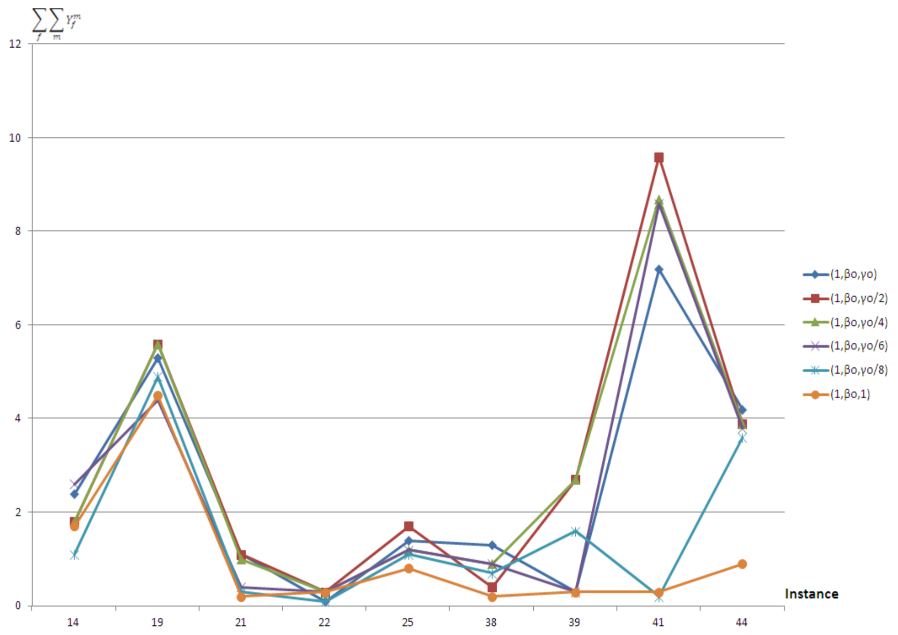


Figure 4.16: Model $IP5$, total number of machine disqualifications for different values of γ

sum of completion times for all preference vectors are obtained (CPU times are below the time limit), it can be noted that, for $\gamma = \gamma_o/2$, $\gamma_o/4$ and $\gamma_o/6$, the sums of completion times are respectively 2916, 2923 and 2921. This variation of the sum of completion times is not strange, since the sum of completion times does not always have to be antagonistic with the expected yield. When scheduling to minimize the sum of completion times, there is a tendency to schedule the same job of a given family on the same machine in order to decrease the number of setups and hence the sum of completion times. This could be an advantage when maximizing the expected yield as well as a disadvantage. Keeping on scheduling the jobs of the same family may lead to a maximized yield if the family has a maximal yield on the machine. On the contrary, if this is not the case, the expected yield may be degraded.

4.4.2 Pareto frontiers for PTC

The search space in the context of multiple objectives can be divided into two non-overlapping regions, namely one which is optimal and one which is non optimal. Although PTC is a two-objective problem, this is also true for

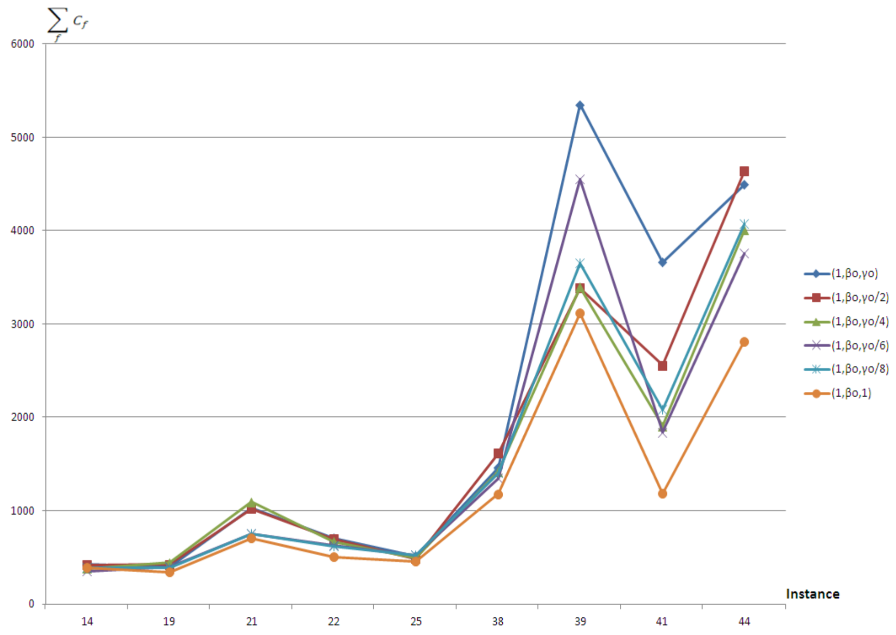


Figure 4.17: Model $IP5$, sum of completion times for different values of γ

$PEHF$ with three objectives. PTC has conflicting objectives, and hence the set of its optimal solutions contains usually more than one solution. These solutions, namely the Pareto-optimal solutions, are of equal importance when there is no previous *higher-level* information. If higher-level information is available, a biased search can be performed.

Let us recall that the objective function of PTC includes two criteria: A classical scheduling criterion, the sum of the completion times of jobs, and the sum of losses in machine qualifications. Thereafter, we see how both criteria interact with each other by fixing a maximum number of possible losses in machine qualifications and minimizing the sum of completion times. This is equivalent to applying the ϵ-constraint method.

The results in Tables 11.a through 11.d show on some selected instance types (14, 19, 21, 22, 25, 38, 39, 41, 42 and 44), the strong trade-off that exists between keeping as many qualifications on machines as possible and completing jobs as soon as possible and its effect on the number of setups and the CPU time.

Figure 4.18 shows, for Instance type 38, the sum of completion times, the sum of losses in machine qualifications, the number of setups and the CPU time. The Pareto frontiers are observed for the two criteria of the objective function ($\sum_f \sum_m Y_f^m$) and $\sum_{f \in F} C_f$). In addition, the number of setups and the CPU time behave in the same way, i.e. they decrease for larger values of allowed losses in machine qualifications. Hence, we find that other aspects are also related and are contradictory.

Moreover, considering Instance type 19 for example, the optimal solution is 297 when the sum of completion times is prioritized, but the thresholds associated to job families on machines must be violated at least 7 times. On the other hand, the global minimum number of disqualifications is 4, with a sum of completion times of 351. The other instance types show as well that the sum of completion times can sharply increase when the maximum number of losses in machine qualifications is reduced and *vice-versa*.

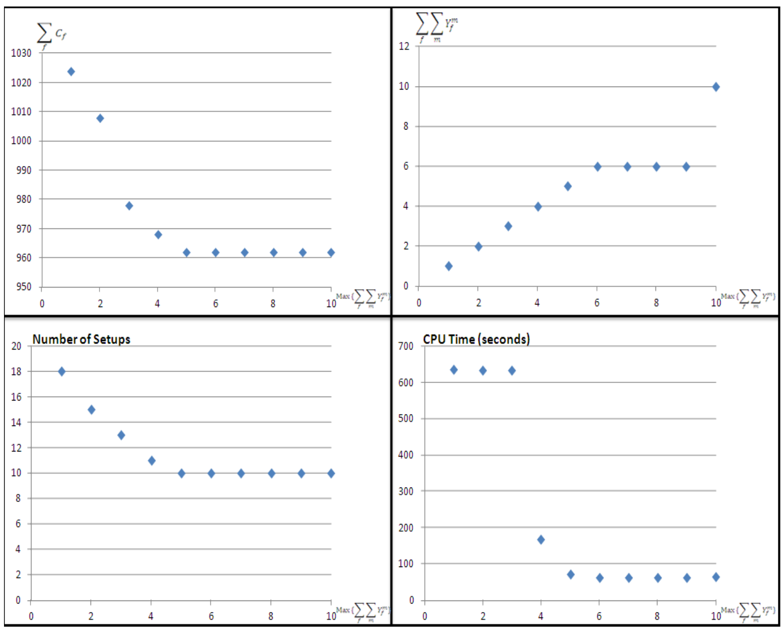


Figure 4.18: Pareto frontiers, Instance type 38

4.4.3 Threshold sensitivity analysis for *PTC*

In this section, we analyze the impact of the time thresholds on the sum of losses in machine qualifications. An instance of the 10 different Instance types is chosen and analyzed. The threshold of the first family is varied from 1 to the time horizon T, while the thresholds of the other families are set to T

to focus on the threshold of one family at a time i.e., $\forall f \in F, f \neq f_1, \gamma_f = T$. The same instances that in Section 4.4.2 are studied. The sum of losses in machine qualifications when varying the threshold are shown in Table 4.12. The tests are done while prioritizing the qualification criterion ($\alpha = 1$ and $\beta = |N| * T$) to study the effect of the threshold on this criterion. The results show that the number of machine disqualifications decreases as the threshold increases. Machines have more time to process the jobs of one family and to shift for other families to be processed. The sum of losses in machine qualifications for one family on all machines varies from one instance to another. For example, Instances 39 and 42 have different maximum values for a threshold interval of $[1, 8]$, which are 3 and 1 respectively. Also, the solution of Instance 19 admits 2 losses in qualifications for a threshold interval of $[11, 12]$. However, incrementing the threshold by only 1 unit of time leads to 0 losses in qualifications for the same instance. Figure 4.19 shows the staircase behavior between the threshold and the sum of losses in machine qualifications.

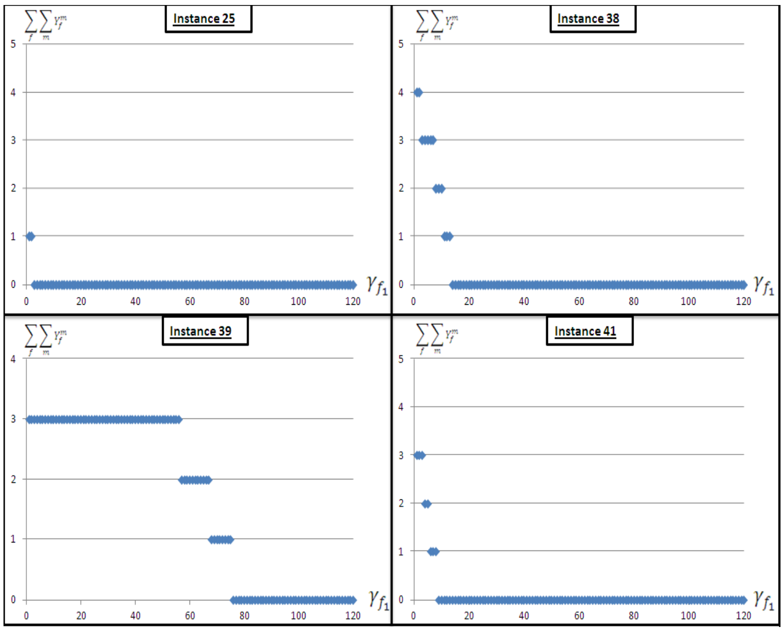


Figure 4.19: Sum of losses in machine qualifications versus threshold variation (Instances of Instance types 25, 38, 39 and 41)

4.5 Conclusion

In this chapter, the results obtained with exact solution approaches were presented and discussed. The results correspond to the problem of scheduling job families on non-identical parallel machines with time constraints (PTC) and the extension to scheduling with Equipment Health Factor ($PEHF$). A multi-criteria objective function is considered, which includes: The sum of completion times, The sum of losses in machine qualifications (PTC,$PEHF$) and the expected yield ($PEHF$).

Exact solutions were determined by using a standard mathematical solver (X-press) based on the family-based mathematical model $IP3$ for PTC and $IP5$ for $PEHF$. The experiments showed that setup times, thresholds, and the number of qualified machines have a direct impact on the complexity of an instance. We noticed that, for instance with a large number of jobs (up to 70 in our instances), the solver cannot get guaranteed optimal solutions in the time limit of 600 seconds. Hence, other solution approaches are required. In Chapter 5, heuristics that target each criterion of the objective function, and numerical results on randomly generated instances are presented. A recursive algorithm and a simulated annealing metaheuristic are also proposed. Numerical results obtained with the developed approaches are compared with the exact solutions given by the standard solver.

Table 10.a: Model $IP5$, ($\alpha^{'} = 1$, $\beta^{'} = |N| * T$, $\gamma_o^{'} = |M| * |F| * |N| * T$), Sum of expected yield ($\sum_{t=1}^{T} \sum_f \sum_m x_{f,t}^m v_{f,t}^m$) and $\sum_f \sum_m Y_f^m$ with variations of γ

	$\gamma^{'}$ - $\sum_{t=1}^{T} \sum_f \sum_m x_{f,t}^m v_{f,t}^m$						$\gamma^{'}$ - $\sum_f \sum_m Y_f^m$					
$No.$	$\gamma_o^{'}$	$\gamma_o^{'}/2$	$\gamma_o^{'}/4$	$\gamma_o^{'}/6$	$\gamma_o^{'}/8$	1	$\gamma_o^{'}$	$\gamma_o^{'}/2$	$\gamma_o^{'}/4$	$\gamma_o^{'}/6$	$\gamma_o^{'}/8$	1
1	922	922	909	896	883	880	0.0	0.0	0.0	0.0	0.0	0.0
2	892	890	890	890	890	888	0.0	0.0	0.0	0.0	0.0	0.0
3	952	952	952	952	952	952	1.3	1.2	1.2	1.1	1.1	0.9
4	844	844	844	844	844	844	2.1	2.1	2.1	1.9	1.9	1.5
5	854	835	835	835	835	835	3.4	2.4	2.3	2.3	1.8	2.7
6	883	882	879	879	879	879	2.1	1.6	0.9	0.9	0.8	0.6
7	909	909	909	909	909	909	1.2	1.1	1.0	1.0	0.9	0.8
8	954	954	954	954	950	939	2.4	2.1	2.1	2.1	2.0	1.6
9	890	890	890	890	890	868	0.2	0.1	0.1	0.2	0.1	0.3
10	924	924	924	924	924	911	0.1	0.2	0.2	0.1	0.1	0.2
11	918	918	918	918	918	905	0.1	0.2	0.2	0.2	0.1	0.2
12	948	948	948	948	948	905	0.1	0.1	0.1	0.0	0.1	0.1
13	939	939	939	939	939	908	0.2	0.2	0.2	0.2	0.1	0.1
14	1853	1840	1837	1840	1837	1837	2.4	1.8	1.8	2.6	1.1	1.7
15	1835	1835	1835	1835	1831	1818	6.2	6.0	6.0	5.9	5.9	5.8
16	1892	1892	1892	1891	1891	1794	0.2	0.3	0.3	0.1	0.1	0.2
17	1820	1820	1820	1820	1820	1815	0.1	0.3	0.2	0.2	0.1	0.2
18	1787	1766	1759	1774	1777	1738	1.2	0.2	0.2	0.2	0.2	0.2
19	1765	1776	1769	1765	1775	1755	5.3	5.6	5.6	4.4	4.9	4.5
20	2748	2748	2748	2745	2739	2698	0.1	0.3	0.3	0.3	0.2	0.1
21	2833	2833	2833	2682	2684	2664	1.1	1.1	1.0	0.4	0.3	0.2
22	2747	2752	2752	2749	2746	2693	0.1	0.3	0.3	0.3	0.1	0.3
23	2761	2733	2725	2737	2724	2679	11.8	9.3	8.2	7.4	8.1	6.6
24	2875	2883	2836	2875	2856	2758	3.2	3.6	2.4	2.7	2.7	1.9
25	2644	2626	2626	2644	2644	2618	1.4	1.7	1.2	1.2	1.1	0.8
26	3936	3936	3936	3936	3936	3746	0.2	0.3	0.2	0.2	0.2	0.2
27	3814	3778	3778	3778	3778	3606	1.3	0.4	0.4	0.4	0.2	0.2
28	3568	3566	3560	3560	3560	3534	0.2	0.2	0.2	0.2	0.2	0.1
29	3583	3570	3572	3612	3591	3584	6.2	5.8	5.7	6.6	5.6	4.6
30	3666	3641	3666	3666	3666	3568	0.1	0.4	0.3	0.3	0.2	0.1
31	4700	4699	4695	4694	4693	4691	0.1	0.3	0.3	0.2	0.1	0.1
32	4194	4210	4194	4194	4194	4194	0.2	0.6	0.4	0.3	0.3	0.2
33	4454	4426	4426	4425	4417	4378	3.4	0.4	0.6	0.3	0.2	0.2
34	4708	4693	4720	4730	4741	4597	2.3	0.8	1.6	1.6	1.6	0.1
35	4608	4608	4608	4608	4608	4346	0.1	0.4	0.4	0.2	0.2	0.1
36	4776	4776	4776	4776	4776	4544	0.2	0.3	0.3	0.3	0.2	0.2
37	4636	4445	4431	4463	4431	4356	7.1	2.8	2.8	2.8	2.6	1.8
38	4561	4514	4545	4542	4559	4470	1.3	0.4	0.9	0.9	0.7	0.2
39	5528	5546	5546	5503	5545	5354	0.3	2.7	2.7	0.3	1.6	0.3
40	5626	5529	5620	5650	5446	5383	9.8	9.8	9.8	9.6	8.1	7.5
41	5787	5667	5522	5556	5595	5324	7.2	9.6	8.7	8.6	0.2	0.3
42	5519	5505	5433	5408	5498	5303	5.4	5.2	5.6	5.6	4.7	0.6
43	6435	6435	6435	6435	6435	6266	0.3	0.3	0.3	0.3	0.2	0.3
44	6242	6122	6128	6111	6137	6099	4.2	3.9	3.9	3.8	3.6	0.9
45	6741	6727	6667	6716	6666	6396	2.3	0.4	0.4	0.4	0.3	0.3
46	6370	6358	6342	6333	6357	6266	4.1	4.2	4.1	4.1	3.9	3.7
47	6418	6492	6364	6370	6369	6200	7.4	7.0	6.9	6.8	6.8	5.8
48	6195	6213	6197	6238	6205	6099	2.1	1.4	1.3	3.8	3.6	1.6
49	-	-	-	-	-	-	-	-	-	-	-	-
50	-	-	-	-	-	-	-	-	-	-	-	-

Table 10.b: Model $IP5$, ($\alpha^{'} = 1$, $\beta^{'} = |N| * T$, $\gamma^{'}_{o} = |M| * |F| * |N| * T$), $\sum_{f \in F} C_f$ and CPU time with variation of γ

	$\gamma^{'}$ - $\sum_f C_f$						$\gamma^{'}$ - (CPU)					
$No.$	$\gamma^{'}_{o}$	$\gamma^{'}_{o}/2$	$\gamma^{'}_{o}/4$	$\gamma^{'}_{o}/6$	$\gamma^{'}_{o}/8$	1	$\gamma^{'}_{o}$	$\gamma^{'}_{o}/2$	$\gamma^{'}_{o}/4$	$\gamma^{'}_{o}/6$	$\gamma^{'}_{o}/8$	1
1	404	404	341	296	267	267	2.1	2.6	2.6	3.1	2.6	2.6
2	311	284	284	284	284	284	2	2.3	1.9	2.2	1.8	1.8
3	669	669	669	669	669	669	3.3	7.7	3	11.5	5	4
4	231	231	231	231	231	231	21.8	24	43	22.9	14.6	15.6
5	278	326	326	332	329	332	600	600	600	600	600	600
6	524	421	356	351	351	351	600	600	600	600	600	600
7	133	133	133	133	133	133	27.6	8.2	6.7	3	3.7	3.8
8	101	101	101	101	92	78	8.3	9	8.4	14	22.5	9.7
9	136	136	136	136	136	116	1.2	1.4	0.8	0.7	0.7	0.3
10	50	50	50	50	50	46	0.1	0.1	0.1	0.1	0.1	0.1
11	114	114	114	114	114	96	1	1.4	0.9	0.9	1	0.5
12	107	107	107	107	107	67	7.8	5.1	9.9	4	3	0.7
13	50	50	50	50	50	34	0.2	0.2	0.2	0.2	0.2	0.2
14	427	425	386	359	396	396	600	600	600	600	600	600
15	349	349	349	349	334	313	245	363.8	599.9	446.9	599.9	600
16	762	762	762	739	739	404	1.1	1.3	1.9	2.3	2.7	3.7
17	176	176	176	176	176	163	4.6	8.1	6	5.7	5.4	3.5
18	708	440	459	430	418	407	600	600	600	600	600	30.8
19	400	428	446	407	392	348	600	600	600	600	600	600
20	1687	1687	1687	1582	1414	1163	1.3	1.2	1.3	1.2	1.4	3.1
21	1033	1024	1097	759	761	707	299.2	190.6	600	600	600	600
22	711	698	670	635	621	512	600	600	600	600	600	310.6
23	1046	895	773	837	808	574	600	600	600	600	600	600
24	1439	1327	1179	1308	1026	666	600	600	600	600	600	600
25	520	492	502	528	525	460	600	600	600	600	600	600
26	2550	2550	2550	2550	2550	1108	9.5	8	4.6	3.5	3.2	1.4
27	3113	2916	2923	2921	2916	2064	57.5	53.6	113.8	96.1	123.8	30.9
28	1908	1840	1322	1322	1322	1135	548.1	600	104	50.5	48	21
29	1960	1946	2000	2559	2362	1776	600	600	600	600	600	600
30	1981	1997	1981	1981	1981	1355	414.1	600	206.2	421.2	128.1	115.2
31	2925	2810	2490	2445	2414	2394	0.8	1	1	1.3	1.3	1.2
32	3763	3379	3763	3776	3766	3765	421.8	600	275.2	600	600	600
33	4708	4301	3688	3267	3108	2752	600	600	600	600	600	600
34	6881	6420	5533	5167	5594	2977	600	600	600	600	600	600
35	4040	4040	4040	4040	4040	1926	33	33	32.6	33	33.2	44
36	2659	2658	2667	2659	2673	1329	600	600	600	600	600	35
37	3636	1887	1969	2031	1976	1403	600	600	600	600	600	600
38	1466	1617	1423	1348	1409	1183	600	600	600	600	600	600
39	5357	3388	3399	4558	3655	3125	600	600	600	600	600	600
40	3876	4610	4445	4485	4311	3281	600	600	600	600	600	600
41	3670	2562	1911	1844	2089	1189	600	600	600	600	600	600
42	2980	2407	2762	1273	1997	1020	600	600	600	600	600	600
43	5076	5102	5026	5077	5045	3825	600	600	600	600	600	600
44	4497	4645	4013	3761	4083	2820	600	600	600	600	600	600
45	6355	6858	4579	6496	4580	2603	600	600	600	600	600	58.9
46	4966	4910	4751	4515	4685	4118	600	600	600	600	600	600
47	7247	7490	5401	6061	6833	4214	600	600	600	600	600	600
48	3763	3716	3375	3554	3734	2531	600	600	600	600	600	600
49	-	-	-	-	-	-	-	-	-	-	-	-
50	-	-	-	-	-	-	-	-	-	-	-	-

Table 10.c: Model $IP5$, ($\alpha^{'} = 1$, $\beta^{'} = |N| * T$, $\gamma^{'}_{o} = |M| * |F| * |N| * T$), Percentage of instance types solved to optimality with variations of γ

	β - % of Opt.					
No.	$\gamma^{'}_{o}$	$\gamma^{'}_{o}/2$	$\gamma^{'}_{o}/4$	$\gamma^{'}_{o}/6$	$\gamma^{'}_{o}/8$	1
1	100 %	100 %	100 %	100 %	100 %	100 %
2	100 %	100 %	100 %	100 %	100 %	100 %
3	100 %	100 %	100 %	100 %	100 %	100 %
4	100 %	100 %	100 %	100 %	100 %	100 %
5	10 %	10 %	10 %	10 %	10 %	10 %
6	10 %	10 %	10 %	10 %	10 %	10 %
7	100 %	100 %	100 %	100 %	100 %	100 %
8	100 %	100 %	100 %	100 %	100 %	100 %
9	100 %	100 %	100 %	100 %	100 %	100 %
10	100 %	100 %	100 %	100 %	100 %	100 %
11	100 %	100 %	100 %	100 %	100 %	100 %
12	100 %	100 %	100 %	100 %	100 %	100 %
13	100 %	100 %	100 %	100 %	100 %	100 %
14	10 %	10 %	10 %	10 %	10 %	10 %
15	100 %	70 %	20 %	20 %	10 %	10 %
16	100 %	100 %	100 %	100 %	100 %	100 %
17	100 %	100 %	100 %	100 %	100 %	100 %
18	10 %	0 %	0 %	0 %	0 %	100 %
19	10 %	10 %	10 %	10 %	10 %	10 %
20	100 %	100 %	100 %	100 %	100 %	100 %
21	100 %	90 %	10 %	10 %	10 %	0 %
22	20 %	0 %	0 %	0 %	0 %	90 %
23	20 %	0 %	0 %	0 %	0 %	0 %
24	10 %	0 %	0 %	0 %	0 %	0 %
25	10 %	0 %	0 %	0 %	0 %	10 %
26	100 %	100 %	100 %	100 %	100 %	100 %
27	100 %	100 %	90 %	100 %	90 %	100 %
28	60 %	0 %	90 %	100 %	100 %	100 %
29	10 %	0 %	0 %	0 %	0 %	0 %
30	70 %	0 %	80 %	60 %	100 %	100 %
31	100 %	100 %	100 %	100 %	100 %	100 %
32	80 %	0 %	90 %	0 %	0 %	0 %
33	0 %	0 %	0 %	0 %	0 %	0 %
34	0 %	0 %	0 %	0 %	0 %	0 %
35	10 %	0 %	0 %	0 %	0 %	100 %
36	0 %	0 %	0 %	0 %	0 %	100 %
37	0 %	0 %	0 %	0 %	0 %	0 %
38	0 %	0 %	0 %	0 %	0 %	0 %
39	0 %	0 %	0 %	0 %	0 %	10 %
40	0 %	0 %	0 %	0 %	0 %	0 %
41	0 %	0 %	0 %	0 %	0 %	0 %
42	0 %	0 %	0 %	0 %	0 %	0 %
43	0 %	0 %	0 %	0 %	0 %	0 %
44	0 %	0 %	0 %	0 %	0 %	0 %
45	0 %	10 %	10 %	10 %	10 %	0 %
46	0 %	0 %	0 %	0 %	0 %	10 %
47	0 %	0 %	0 %	0 %	0 %	0 %
48	0 %	0 %	0 %	0 %	0 %	0 %
49	-	-	-	-	-	-
50	-	-	-	-	-	-

Table 11.a: Model $IP3$, ($\alpha = 1$, $\beta = 1$), Maximum sum of losses in machine qualifications ($(\sum_f \sum_m Y_f^m)_{Max}$) versus $\sum_{f \in F} C_f$

	Instance type - $\sum_f C_f$									
$(\sum_f \sum_m Y_f^m)_{Max}$	14	19	21	22	25	38	39	41	42	44
0	-	-	707	512	-	-	3131	1194	-	3119
1	398	-	607	494	467	1024	3048	1180	1032	2952
2	368	-	559	486	456	1008	3018	1165	881	3139
3	346	-	559	483	435	978	2992	1160	865	2731
4	346	351	559	483	429	968	2992	1158	852	2714
5	346	329	559	483	429	962	2992	1158	842	2711
6	346	306	559	483	427	962	2992	1158	834	2711
7	346	297	559	483	427	962	2992	1158	834	2711
8	346	297	559	483	427	962	2992	1158	834	2711
9	346	297	559	483	427	962	2992	1158	834	2711
10	346	297	559	483	427	962	2992	1158	834	2711

Table 11.b: Model $IP3$, ($\alpha = 1$, $\beta = 1$), Maximum sum of losses in machine qualifications ($(\sum_f \sum_m Y_f^m)_{Max}$) versus $\sum_f \sum_m Y_f^m$

	Instance type - $\sum_f \sum_m Y_f^m$									
$(\sum_f \sum_m Y_f^m)_{Max}$	14	19	21	22	25	38	39	41	42	44
0	-	-	0.0	0.1	-	-	0.1	0.0	-	0.0
1	0.9	-	0.9	0.8	0.9	0.8	0.8	0.9	0.9	0.8
2	1.9	-	1.8	1.8	1.8	1.8	1.7	1.8	1.9	1.9
3	2.8	-	1.8	2.9	2.8	2.7	2.9	2.8	2.9	2.9
4	3.9	3.9	3.9	3.8	4.0	3.8	3.7	3.9	3.8	3.8
5	3.8	4.8	4.3	4.4	4.9	4.6	3.4	4.1	4.9	4.8
6	3.8	5.9	4.1	4.3	5.7	5.8	3.4	4.4	5.8	5.8
7	3.8	6.8	6.8	4.3	5.7	6.3	3.3	6.7	6.8	6.9
8	7.9	7.9	6.7	7.8	5.8	6.4	3.4	7.6	7.7	7.2
9	7.8	8.9	6.6	7.8	5.7	6.2	8.8	8.8	7.6	7.1
10	7.9	9.6	6.7	7.7	5.9	9.8	8.9	9.7	7.6	7.1

Table 11.c: Model $IP3$, ($\alpha = 1$, $\beta = 1$), Maximum sum of losses in machine qualifications versus total number of setups

	Instance type - Number of Setups									
$(\sum_f \sum_m Y_f^m)_{Max}$	14	19	21	22	25	38	39	41	42	44
0	-	-	19	10	-	-	11	14	-	28
1	11	-	12	9	15	18	10	13	36	22
2	8	-	6	6	15	15	8	12	19	23
3	5	-	6	6	11	13	7	11	23	13
4	5	12	3	6	9	11	7	11	16	11
5	5	9	6	6	11	10	7	11	20	9
6	5	7	6	6	8	10	5	11	19	9
7	5	6	6	6	8	10	5	11	18	4
8	5	6	6	6	8	10	3	11	20	4
9	5	6	6	6	8	10	7	11	20	4
10	5	6	6	6	8	10	7	11	20	4

Table 11.d: Model $IP3$, ($\alpha = 1$, $\beta = 1$), Maximum sum of losses in machine qualifications versus CPU time

	Instance type - CPU									
$(\sum_f \sum_m Y_f^m)_{Max}$	14	19	21	22	25	38	39	41	42	44
0	-	-	244.5	138.7	-	-	600	600	-	600
1	600	-	16.6	41.9	600	600	600	600	600	600
2	105.7	-	5.8	9.9	600	600	600	333.6	600	600
3	9.8	-	7.1	9.8	104.8	600	136.7	119.1	600	600
4	12.9	600	6.7	7.8	39.7	168.7	119.4	95.3	165.8	283
5	10.6	600	6.5	7.2	35	72.4	135.1	91.7	50.3	199.2
6	10.6	338.5	6.3	7.3	28.8	62.7	132	91.7	42.6	185.4
7	10.7	36.2	4.2	7.1	28.6	63.3	133.2	96.1	42.5	183.3
8	8.5	35.8	4.2	6.7	28.8	63	129.8	96.4	41.5	183.4
9	8.2	32.8	4.1	6	28.7	63.3	36.5	95.6	41.7	184.6
10	8.4	31.7	4.1	7	28.9	64.6	36.4	101.8	41.7	184.1

Table 11.e: Model $IP3$, ($\alpha = 1$, $\beta = 1$), Maximum sum of losses in machine qualifications versus percentage of optimal solution

	Instance type - % of Opt.									
$(\sum_f \sum_m Y_f^m)_{Max}$	14	19	21	22	25	38	39	41	42	44
0	-	-	90 %	100 %	-	-	0 %	0 %	-	0 %
1	10 %	-	100 %	100 %	0 %	0 %	0 %	0 %	10 %	10 %
2	100 %	-	100 %	100 %	10 %	0 %	10 %	80 %	10 %	10 %
3	100 %	-	100 %	100 %	100 %	10 %	100 %	100 %	10 %	10 %
4	100 %	10 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	90 %
5	100 %	10 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	90 %
6	100 %	90 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %
7	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %
8	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %
9	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %
10	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %

Table 4.12: Model $IP3$, ($\alpha = 1$, $\beta = |N| * T$), Threshold sensitivity study

	$\sum_f \sum_m Y_f^m$				
$No.$	4	3	2	1	0
14	-	-	-	[1-6]	[7-94(T)]
19	[1-2]	[3-10]	[11-12]	-	[13-112(T)]
21	-	-	[1-2]	[3-8]	[9-118(T)]
22	-	-	[1-2]	[3-6]	[7-92(T)]
25	-	-	-	[1-2]	[3-140(T)]
38	[1-2]	[3-7]	[8-10]	[11-13]	[14-191(T)]
39	-	[1-56]	[57-67]	[68-75]	[76-346(T)]
41	-	[1-3]	[4-5]	[6-8]	[9-199(T)]
42	-	-	-	[1-8]	[9-153(T)]
44	-	[1-34]	[35-42]	[43-64]	[65-281(T)]

Chapter 5

Heuristics and Metaheuristics

5.1 Introduction

Heuristics and a metaheuristic for PTC and $PEHF$ are presented in this chapter. Numerical experiments are conducted and the results are compared and analyzed. First, a List Heuristic (LH) is provided which tries to schedule jobs by earliest job family threshold. A Scheduling-Centric Heuristic (SCH) is then proposed. It aims at minimizing the sum of completion times by trying to reduce the number of setups while scheduling job families. Regarding machine qualifications, a Qualification-Centric Heuristic (QCH) is presented that aims at minimizing the number of losses in machine qualifications by trying to satisfy time thresholds each time a job is scheduled on any machine. A Yield-Centric Heuristic (YCH) is proposed for $PEHF$. The idea of this heuristic is to maximize the sum of expected yield by assigning jobs of families to machines with a maximal yield.

Further, an Recursive Heuristic (RH) is introduced and applied on the previous constructive heuristics. RH aims at finding solutions with a minimized sum of losses in machine qualifications. This is done by testing the final solution obtained by a given constructive heuristic for losses in machine qualifications, and then re-applying the same heuristic after disqualifying one or more machines on one or more families. Finally, a Simulated Annealing (SA) is adapted to handle PTC and $PEHF$. SA is applied on the solutions obtained from RH. Results on the constructive heuristics, recursive heuristics, and SA metaheuristic are provided and compared to the exact solutions of Chapter 4.

The chapter is structured as follows. In Section 5.2, constructive heuristics for both problems are described. Section 5.3 addresses the recursive heuristic. Simulated Annealing is proposed in Section 5.4. Section 5.5 presents the numerical results obtained for PTC and Section 5.6 for $PEHF$. Section 5.7 concludes the chapter.

5.2 Constructive Heuristics

Various heuristics are developed to solve PTC and $PEHF$: A List Heuristic(LH), a Scheduling-Centric Heuristic(SCH) and a Qualification-Centric Heuristic(QCH) heuristics for PTC, and a Yield-Centric Heuristic (YCH) for $PEHF$.

5.2.1 List Heuristic (LH)

In the list heuristic, jobs of different families are scheduled on parallel machines by combining two priority rules in the following order:

1. Earliest Time Threshold,
2. Shortest Processing Time (SPT).

The combined rules described in Algorithm 1. The Earliest Time Threshold rule schedules jobs of the family with the earliest threshold, i.e. the most urgent family (has the highest priority compared to other jobs of other families) to keep the machine qualified. In case of equal thresholds, the selection is performed using the SPT rule.

The mechanism of LH is described in the pseudo-code of Algorithm 2. The function $DISQUALIFY$ allows a machine to be disqualified for a certain family when the qualification could not be maintained either because of a threshold violation or because there are no jobs to schedule. LH is sketched below.

- For each machine, schedule first a job of the qualified family with the smallest threshold.
- While there are still jobs to schedule in N,
  - For each machine m in M,
    - If the current family f on m is non empty and family f has the smallest threshold, then

Algorithm 1 Earliest Time Threshold

```
1: $f = 1$
2: for $f = 1 \rightarrow |F|$ do
3:    if $n_f \neq 0$ then
4:       if $m(f) = 1$ then
5:          if $f_m^* = 0$ then
6:             $f_m^* = f$
7:          else
8:             if $\gamma_f < \gamma_{f_m^*}$ then
9:                $f_m^* = f$
10:            else
11:               if $\gamma_f = \gamma_{f_m^*}$ then
12:                  if $p_f < p_{f_m^*}$ then
13:                     $f_m^* = f$
14:                  end if
15:               end if
16:            end if
17:         end if
18:      end if
19:   end if
20: end for
```

  · If there is a job in f that can be scheduled on m, then schedule a job of f on m,
  · Else, select the next non-empty qualified family f with the smallest threshold on m, and schedule a job of f on m.

∗ Else, select the next non-empty qualified family f with the smallest threshold on m, and schedule a job of f on m.

The time complexity of LH is $\mathcal{O}(|N||M||F|)$, since the main loop is performed at most $|N|$ times, the second loop $|M|$ times, and at most $|F|$ families are checked each time a job is scheduled. The practical complexity will be much lower since $|M|$ jobs will often be scheduled in each main loop.

5.2.2 Scheduling-Centric Heuristic (SCH)

The main goal of the Scheduling-Centric Heuristic (SCH) is to minimize setup times. Recall that a setup time is necessary when two jobs of different families are scheduled on a machine consecutively. The heuristic schedules

Algorithm 2 List Heuristic

```
1: $i = 1$
2: for $m = 1 \rightarrow |M|$ do
3:     $f_m^* = 0$
4: end for
5: while $i \leq |N|$ do
6:     $Feasible = FALSE$
7:     $m = 1$
8:     for $m = 1 \rightarrow |M|$ do
9:         - Select $f_m^*$ by applying the Earliest Time Threshold algorithm
10:        if $f_m^* \neq 0$ then
11:            - Update $C_{f_m^*}$ by taking into considerations setup time if necessary
12:            $n_{f_m^*} = n_{f_m^*} - 1$
13:            $i = i + 1$
14:            $Feasible = TRUE$
15:        else
16:            $DISQUALIFY(m, f_m^*)$
17:            $y_{f^*}^m = y_{f^*}^m + 1$
18:        end if
19:    end for
20:    if $Feasible = FALSE$ then
21:        print $No\ feasible\ solution.$
22:    end if
23: end while
```

jobs of the same family successively on a machine until either all jobs of the family are scheduled or it is no longer possible to schedule a job of the family without losing the qualification of another family on the machine. If it is no longer possible to schedule a job of a family before its remaining time threshold, then the machine is disqualified for the family.

SCH is sketched below, where a family f is called "non empty" if there is at least one job to schedule in f, and is called "current" for a machine m if f is the family of the last job scheduled on m. The "remaining time threshold" for a qualified family f on machine m corresponds to the time between the end of the last job scheduled on m and the latest allowed start time of a job of f before losing its qualification on m. If the remaining time threshold is negative, then family f becomes disqualified on m. This could happen either because there is not enough time to schedule a job of f on m or because all jobs of f are already scheduled.

- For each machine, schedule first a job of the qualified family with the smallest threshold.

- While there are still jobs to schedule in N,
  - For each machine m in M,
    - If the current family f on m is non empty, then
      - If there is a job in f that can be scheduled on m without losing the qualification of a non-empty family, then schedule a job of f on m.
      - Else, select the non-empty qualified family f with the smallest positive remaining threshold on m, and schedule a job of f on m.
    - Else, select the non-empty qualified family f with the smallest positive remaining threshold on m, and schedule a job of f on m.

This heuristic is implemented following the pseudo-code in Algorithm 3, where the function MIN_NEXT aims at selecting the family with the minimum threshold to be scheduled next to the family in process and $UPDATE\_TIMES$ is a time sweeper that updates start and end times on each machine.

Algorithm 3 SCH pseudo code

```
1:  $i = 1$
2:  for $m = 1 \to |M|$ do
3:      $IDLE_m = TRUE$
4:      $t_m = 0$
5:      $f^*_m = 0$
6:  end for
7:  while $i \leq |N|$ do
8:      $Feasible = FALSE$
9:      $m = 1$
10:     - Select the family with the minimum threshold and that can be processed
        on $m$ ($f^*_m$)
11:     for $m = 1 \to |M|$ do
12:         if $IDLE_m = TRUE$ then
13:             if $f^*_m \neq 0$ then
14:                 $min\_next = MIN\_NEXT(\gamma_F, f^*_m)$
15:                 if $t_m < min\_next$ then
16:                     if $n_{f^*_m} \neq 0$ then
17:                         $IDLE_m = FALSE$
18:                         $C_{f^*_m} = UPDATE\_TIMES(f^*_m)$
19:                         $t_m = t_m + p_{f^*_m}$
20:                         $n_{f^*_m} = n_{f^*_m} - 1$
21:                         $Feasible = TRUE$
22:                         $i = i + 1$
23:                     end if
24:                 else
25:                     - Select $f^*_m$ with $n_{f^*_m} > 0$ and with shortest threshold
26:                     $t_m = t_m + s_{f^*}$
27:                 end if
28:             else
29:                 $DISQUALIFY(m, f^*_m)$
30:                 $y^m_{f^*_m} = y^m_{f^*_m} + 1$
31:             end if
32:         end if
33:     end for
34:     for $m = 1 \to |M|$ do
35:         $IDLE_m = TRUE$
36:     end for
37:     if $Feasible = FALSE$ then
38:         print $No\ feasible\ solution.$
39:     end if
40: end while
```

The time complexity of SCH is also $\mathcal{O}(|N||M||F|)$, since the main loop is performed at most $|N|$ times, the second loop $|M|$ times, and at most $|F|$ families are checked each time a job is scheduled. The practical complexity will be much lower since $|M|$ jobs will often be scheduled in each main loop, and the same family will often be selected.

5.2.3 Qualification-Centric Heuristic (QCH)

The main objective of the Qualification-Centric Heuristic is to minimize the total number of violations of the time constraint on the machines on which each family is qualified. Hence, in the first phase of QCH, jobs are scheduled on a machine by always prioritizing the family qualified on the machine with the smallest remaining time threshold. However, this usually leads to a large number of setups. This is why two other phases of local improvements are applied where no additional machine disqualifications are allowed. Phase 2 tries to advance the last job of each machine by combining it with the first job of the same family scheduled on the machine, i.e. only intra-machine changes are evaluated. Phase 3 allows inter-machine changes, i.e. the last job or group of jobs of the same family are combined with a job of the same family on another machine.

The heuristic is sketched below, where the notions of "non empty" family and "remaining time threshold" are the same than for the Scheduling-Centric Heuristic (SCH) described in the previous section. Note that qualifications can be lost since a lot of time is spent in performing setups from one family to another, and there might not be enough jobs in a family to maintain the qualification on a machine on the entire horizon.

- *Phase 1.* While there are still jobs to schedule in N,
  - *Phase 1.* For each machine m in M,
    - ∗ *Phase 1.* Select the non-empty qualified family f on m with the shortest positive remaining threshold, and schedule a job of f on m.
- *Phase 2.* For each machine m in M,
  - *Phase 2.* While a move is performed,

    * *Phase 2.* If scheduling the last job scheduled on m with the first job of the same family scheduled on m does not result in lost qualifications, then perform the move.

- *Phase 3.* For each machine m in M,
  - *Phase 3.* While a move is performed,
    * *Phase 3.* For each machine m' ($m' \neq m$) in M,
      · *Phase 3.* If scheduling the last job (or group of jobs of the same family) scheduled on m with a job of the same family scheduled on m' does not result in lost qualifications, then perform the move.

This heuristic is implemented following the pseudo-code in Algorithm 4 where three more functions are added: $UPDATE\_THRESHOLD$ that updates all family thresholds on the qualified machines once any job is completed and $INTRACHANGE$ and $INTERCHANGE$ that try to shift jobs on the same machine and between machines by moving jobs of the same families together in order to decrease the number of setups. This is done while guaranteeing that no more machine qualification is lost when moving jobs.

The time complexity of Phase 1 is $\mathcal{O}(|N||M||F|)$, of Phase 2 is $\mathcal{O}(|M||N|^2)$ and of Phase 3 is $\mathcal{O}(|M|^2|N|^2)$. Hence, the overall time complexity of QCH is the one of Phase 3, i.e. $\mathcal{O}(|M|^2|N|^2)$. As for LH and SCH, the practical complexity will be much lower, since $|M|$ jobs will often be scheduled in the main loop of Phase 1, and much less changes than the maximum possible will actually be performed in Phases 2 and 3.

5.2.4 Comparing the heuristics related to problem PTC

Table 5.1 provides results for the PTC constructive heuristics. Comparing the heuristics, we notice that SCH is in most cases better than LH and QCH in terms of the sum of completion times (38 out of 47 instance types). On the other hand, as expected, QCH gives the best results in terms of losses in machine qualifications (49 out of 49 instance types). The CPU times are almost the same for all heuristics since solutions are obtained instantaneously. It is important to note that heuristics cannot always solve the instances because, depending on the heuristic, a machine can lose its qualification early for a given family f and the remaining jobs of f may not be scheduled on another machine (no other qualified machine), leading to an infeasible solution. This case is found in Instance type 6 for LH and QCH and in Instance types 6, 23 and 29 for SCH.

Table 5.1: Comparison between LH, SCH and QCH ($\sum_{f\in F} C_f$ and $\sum_f \sum_m Y_f^m$), best solution in bold

	$\sum_{f\in F} C_f$			$\sum_f \sum_m Y_f^m$			CPU-T		
No.	LH	SCH	QCH	LH	SCH	QCH	LH	SCH	QCH
1	**255**	**255**	**255**	0.4	0.4	**0.3**	0	0	0
2	**245**	**245**	303	0.9	0.9	**0.4**	0	0	0
3	**669**	**669**	**669**	0.9	0.9	**0.8**	0	0	0
4	290	**263**	290	2.8	2.8	**2.7**	0	0	0
5	**323**	**323**	**323**	2.9	2.9	**2.8**	0	0	0
6	-	-	-	-	-	-	-	-	-
7	**153**	**153**	189	2.8	2.8	**1.7**	0	0	0
8	**91**	96	99	4.8	4.7	**2.8**	0	0	0
9	**129**	**129**	165	1.2	1.2	**0.4**	0	0	0
10	**52**	**52**	92	2.1	1.9	**0.3**	0	0	0
11	**123**	**123**	141	1.1	1.0	**0.4**	0	0	0
12	**80**	**80**	**80**	2.2	2.1	**1.8**	0	0	0
13	**34**	**34**	**34**	**0.3**	**0.3**	**0.3**	0	0	0
14	**391**	393	436	3.8	3.7	**2.8**	0	0	0
15	359	**335**	**335**	6.9	**6.7**	**6.7**	0	0	0
16	**404**	**404**	**404**	0.4	**0.3**	**0.3**	0	0	0
17	**167**	**167**	**167**	2.1	2.1	**1.8**	0	0	0
18	**473**	**473**	503	2.8	2.8	**1.9**	0	0	0
19	**340**	**340**	378	7.7	7.7	**5.8**	0	0	0
20	**1182**	**1182**	1239	0.4	0.4	**0.3**	0	0	0
21	**566**	**566**	745	1.9	1.9	**0.9**	0	0	0
22	574	550	**544**	2.2	1.8	**0.8**	0	0	0
23	**690**	-	707	8.2	-	**7.7**	0	-	0
24	**660**	673	859	2.9	2.8	**1.8**	0	0	0
25	497	**496**	679	6.2	6.8	**1.9**	0	0	0
26	**1148**	**1148**	1572	**0.4**	**0.4**	**0.4**	0	0	0
27	**2194**	**2194**	2429	1.2	1.2	**0.8**	0	0	0
28	1671	1671	**1210**	0.8	0.7	**0.4**	0	0	0
29	**2486**	-	2924	5.3	-	**4.1**	0	-	0
30	**1746**	**1746**	2007	0.4	0.4	**0.3**	0	0	0
31	**4133**	**4133**	**4133**	**0.4**	**0.4**	**0.4**	0	0	0
32	**3685**	**3685**	**3685**	1.1	1.1	**1.0**	0	0	0
33	**2765**	**2765**	2947	2.9	2.9	**2.7**	0	0	0
34	3452	3452	**3075**	1.2	1.2	**0.8**	0	0	0
35	**2440**	**2440**	**2440**	**0.4**	**0.4**	**0.4**	0	0	0
36	1861	1913	**1851**	0.4	**0.3**	**0.3**	0	0	0
37	**1615**	**1615**	1957	2.2	2.2	**1.8**	0	0	0
38	1137	**1128**	1301	5.8	5.7	**3.7**	0	0	0
39	**3823**	**3823**	5200	1.2	1.2	**0.8**	0	0	0
40	**3064**	3074	3935	6.8	7.2	**4.4**	0	0	0
41	1304	**1280**	1568	4.1	3.8	**1.9**	0	0	0
42	878	**873**	1055	3.9	3.7	**2.8**	0	0	0
43	3876	**3800**	4593	2.2	2.9	**1.7**	0	0	0
44	3329	**3189**	3591	4.8	4.9	**2.8**	0	0	0
45	**3001**	**3001**	3223	1.2	1.2	**0.9**	0	0	0
46	3715	**3698**	4249	4.4	**4.2**	**4.2**	0	0	0
47	3650	3650	**5634**	3.4	3.4	**2.3**	0	0	0
48	**2963**	**2963**	3427	2.1	2.1	**1.7**	0	0	0
49	2876	**2870**	3074	3.4	3.3	**2.2**	0	0	0
50	3150	**2952**	3582	4.4	4.2	**3.1**	0	0	0

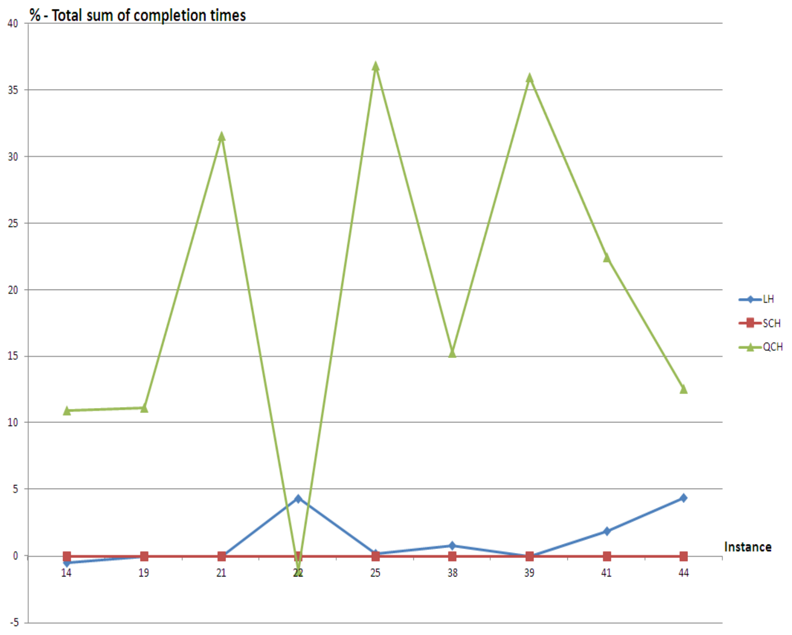


Figure 5.1: Comparison between LH, SCH and QCH - Sum of completion times

Figure 5.1 shows the percentage of increase in the sum of completion times for the constructive heuristics. This percentage is calculated relative to the values obtained by SCH. SCH shows a tendency to minimize the sum of completion times in the majority of the selected instances. Let us recall that these instances are the same as those of Chapter 4. In addition, LH dominates QCH except for Instance type 22 where QCH dominates both LH and SCH. The difference between QCH and SCH can attain 37% in Instance type 25.

The sum of losses in machine qualifications for the constructive heuristics is also shown in Figure 5.2. QCH dominates the other heuristics on all the instance types. There is no dominance between LH and SCH since LH sometimes dominates SCH as in Instance type 25 and SCH sometimes dominates LH as in Instance type 41. Moreover, in Instance type 25, the difference in losses in machine qualifications is around 5 (this corresponds to the largest difference in this figure). This is consistent with the results in Figure 5.1, where the difference in the sum of completion times is the largest as well (around 37%). This illustrates once again the antagonistic nature of both criteria in PTC.

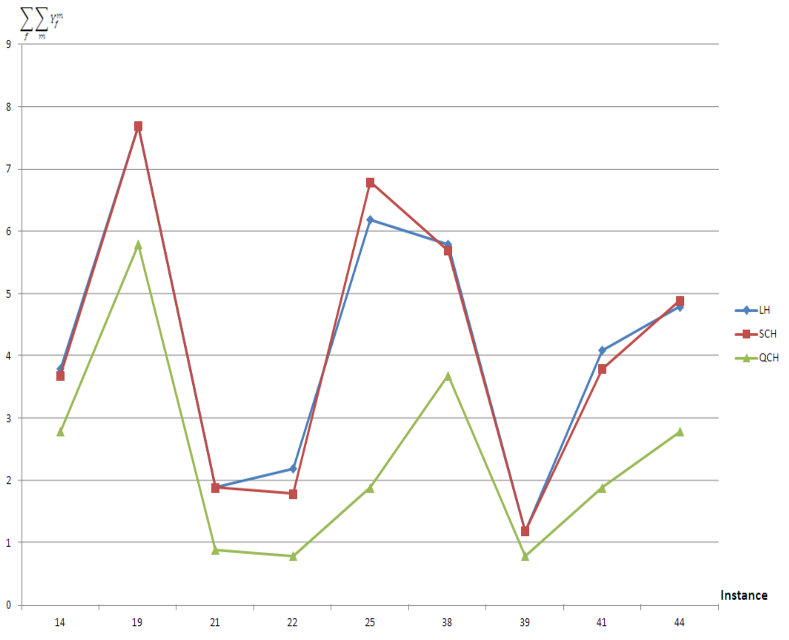


Figure 5.2: Comparison between LH, SCH and QCH - Sum of losses in machine qualifications

5.2.5 Yield-Centric Heuristic (YCH)

The Yield Centric Heuristic (YCH) aims at maximizing the total yield ($\sum_{f\in F}\sum_{m\in M}\sum_{t=1}^{T} x_{f,t}^m v_f^m$)) and is sketched below. It first schedules jobs on the machine with maximum yield. Then, the heuristic tries to guarantee that time thresholds are not violated. It chooses jobs of the same family when the yield is the same and no time threshold is violated.

- For each machine, schedule first a job of the qualified family with the **largest yield**.
- While there are still jobs to schedule in N,
  - For each machine m in M,
    - ∗ If the current family f on m is non empty, then
      - · If there is a job in f that can be scheduled on m without losing the qualification of a non-empty family, then schedule a job of f on m.
      - · Else, select the non-empty qualified family f with the **maximum yield** on m, and schedule a job of f on m.

    * Else, select the non-empty qualified family f with the **maximum yield** on m, and schedule a job of f on m.

This heuristic does not consider the minimum threshold related to a given family, but the yield resulting from assigning a given family on a machine. YCH tries to select jobs of families with maximum yield when assigning these jobs to a given machine. YCH keeps on selecting the same job family while this family is not empty and the assignment of a job of this family does not lead to a machine disqualification. By doing this, the number of setups may be minimized since YCH tends to schedule the same family on the best machine. This may lead to minimizing the sum of completion times. Moreover, machine qualifications are considered in the same way than SCH, where it is checked whether scheduling a job of a job family violates the time threshold of another family and hence causes a loss in machine qualification. The pseudo code corresponding to this heuristic is provided in Algorithm 5. The time complexity is the same as that of SCH, i.e. $\mathcal{O}(|N||M||F|)$.

5.3 Recursive Heuristic (RH)

The general idea of this algorithm is to schedule jobs by accepting each time one qualification loss or more. More precisely, we consider a solution obtained by any of the previous heuristics, and reapply this heuristic after changing the initial qualification scheme. The perturbations in the qualification scheme are chosen from the set of machines that lost their qualifications in the solution. In other words, after scheduling the jobs using a given heuristic, we examine the resulting solution to verify whether the machines are still capable (qualified) to process the jobs, i.e. whether thresholds are satisfied or not. If this is not the case, we change the data to accept some threshold violations, and reapply the heuristic recursively to improve the solution.

- If the solution obtained by the constructive heuristic admits one or more losses in machine qualifications, then
    - For each disqualification on machine m for family f in the solution,
        * Disqualify machine m for family f.
        * Apply the constructive heuristic with the new qualification scheme.
        * If the solution is feasible, then accept it.

  * Else, return the previous solution.

- Else, return the previous solution.

The pseudo code of this method is presented in Algorithm 6. The function $SOLVE$ returns a solution obtained by a given heuristic on a given instance. Each solution s admits 0 or more losses in machine qualifications. The total number of losses in machine qualifications represents the stopping criterion. The function $BINARY$ transforms the integer counter of base 10 into a number of base 2. This guarantees that all possible combinations of machine disqualifications are covered by the counter. The score of a solution $s.SCORE$ is based on the minimum sum of losses in machine qualifications prior to the sum of completion times for PTC and prior to the sum of expected yield and of completion times for $PEHF$.

5.4 Metaheuristic: Simulated Annealing (SA)

Different approaches for multi-objective scheduling problems can be found in the literature as discussed in Chapter 3. Each approach has its own advantages and drawbacks as described in Deb (2001). A need for a general method able to treat a large class of models and independent of the considered objectives shows up. Effectively, metaheuristics, such as Simulated Annealing (SA), have demonstrated their ability to solve combinatorial problems such as production scheduling (Teghem (2002)). Also, some authors suggested to adapt metaheuristics in order to solve multi-objective combinatorial problems (e.g. Ehrgott and Gandibleux (2000)). Since scheduling problems are also combinatorial problems, applying metaheuristics to production scheduling (e.g. scheduling in semiconductor manufacturing) with multiple criteria is suitable. Let us recall that in the studied problems (PTC and $PEHF$) of this thesis, the objectives are conflicting. Thus, a solution may perform well for one objective, but give *bad* results for others. Therefore, any proposed scheduling approach has to find the right trade-off.

The limitations related to exact methods ($IP3$ and $IP5$) regarding the maximum number of jobs, machines and families as shown in Chapter **??** led to the need for a more flexible method that can deal with large scale instances. SA was chosen for this objective. Different parameters of SA such as initial temperature, cooling factor, and number of iterations at each temperature, described in the following section, are problem based. In this section, SA is considered to tackle both PTC and $PEHF$. Experimental results, based on solutions obtained from the previous heuristics after applying the recursive algorithm, are provided in the next sections and real industrial data are also used (see Appendix A). The scoring function of SA is considered each time as a linear combination of the studied criteria while prioritizing one of the criteria over the other depending on the objective of the heuristic. For example, a preference vector (1,1) is used by the scoring function of SA applied on SCH, called in the sequel $SABSCH$, since the final objective of SCH is to minimize the sum of completion times.

5.4.1 Mechanism

Simulated Annealing (SA) is motivated by analogy to annealing in solids. The idea of SA comes from a paper published by (Metropolis et al. (1953)), where an algorithm is developed that simulates the cooling of material in a heat bath. This is a process known as annealing. After heating a solid, passing melting point and then cooling it, the resulting structural properties of the solid are found to be dependent on the rate of cooling. If the liquid is cooled slowly enough, large crystals will be formed. However, if the liquid is cooled quickly (*quenched*), the crystals will contain imperfections. The developed algorithm simulated the material as a system of particles. The algorithm simulates the cooling process by gradually lowering the temperature of the system until it converges to a steady, frozen state.

Simulated annealing was applied to optimization problems to search for feasible solutions and converge to an optimal solution (Kirkpatrick et al. (1983)). In a simulated annealing meta-heuristic algorithm, we use an initial solution to generate a set of neighborhood solutions. We look in the later for a solution which has a *cost/score* less/greater than the cost/score of the initial solution. In order to find the optimal or at least an improved solution, we need to explore the space of solutions in an effective way since the number of solutions is usually enormous (for example, the solution subspace of an initial solution provided by any of the constructive heuristics is of size $|N|^2$ solutions). The exploration of the solution space is done in a simulated annealing algorithm by using two major parameters which are: Temperature, and number of iterations at each temperature. Actually, when exploring the solution space, we may step toward a solution of higher cost from a solution of lower cost. Hence, if the objective function is to minimize a criterion, as in the case of PTC, then it should be ignored in a normal case. However, in a simulated annealing algorithm, a worse solution is accepted with a probability that is defined as a function of the annealing temperature mentioned earlier.

The acceptance of worse solutions helps to leave local minima in case of *cost* minimization (e.g. the objective function of PTC) as well as local maxima in case of cost maximization (e.g. the objective function of $PEHF$). The pseudo code of a basic simulated annealing algorithm is given in Algorithm 7.

The number of iterations at each temperature is determined experimentally. The temperature itself, and consequently the probability of acceptance, is a key factor in the effective exploration of the solution space, and this is called the intensification. In addition, solution neighborhoods are critical in determining the optimal solution. A solution space may be divided into subspaces associated to qualification schemes. We can look at each set of neighborhood solutions as a subspace. If we are looking for an optimal solution (local and may be global) in a set/subspace of neighborhood solutions of a given initial solution, it is possible that we will not find it because the global optimal solution is found in another subspace. Diversification is used to overcome this difficulty. It is based on exploring the solution space each time from a different start point (subspaces). If we consider that our solution space is divided into subspaces related to machine qualification schemes, the diversification can consequently be done by starting with initial solutions of different qualification schemes. This can actually be implemented by starting the SA with different initial solutions. For example, the set of feasible solutions of RH with different qualification schemes may be considered to be the set of initial solutions of SA. This is not treated in the scope of this thesis.

5.4.2 SA for PTC and $PEHF$

While using the simulated annealing algorithm in our search for an optimal solution, we notice that the possibility of finding the optimal solution is strongly related to the initial solution as well as the set of its neighborhoods. Let us recall that our problem is the scheduling of job families on non-identical parallel machines, where non-identical machines means in this case that the machines have different qualification configurations, i.e. not all machines are available to process all job families. The objective function for PTC is bi-criteria. Basically, the initial solution is provided by one of the proposed heuristics. The set of neighborhood solutions of the generated initial solution is determined by the intra-change of job families on each machine and by the inter-change of job families between machines. These are explained later in this section.

The initial solution, which is generated for the simulated annealing algorithm by one of the recursive constructive heuristics, may have a different qualification scheme than the original one. If not, then we have an optimal solution regarding the sum of losses in machine qualifications ($\sum_{f\in F}\sum_{m\in M} Y_f^m$). On the other hand, if there are losses in the machine/family qualifications, then the less the number of losses, the better the solution on the same criterion. In our simulations, the objective function is considered always as a combination of both criteria for PTC and for the three criteria for $PEHF$. Hence, the results for the instance types are comparable to those of the exact solution for the same weights of criteria (preference vector) of the objective function. In our simulated annealing algorithm, the diversification of the solutions while minimizing the sum of completion times, could be defined as the set of feasible solutions with different configurations of machine qualifications.

Using the same instance types, a simulated annealing algorithm is developed and adapted in the next section, and tests are done. The time limit per instance were set to 600 seconds as one of the stopping criteria. The algorithm is tested using the same objective functions that are studied for the exact solutions. The minimum temperature and the number of iterations done at each temperature are determined experimentally and specified below.

Preliminaries

- A weighting function is chosen, the effect of this choice on the procedure is small due the stochastic character of the method. The weighted sum is a well known method and it is the easiest function to compute. In the sequel, the weighting function is computed with preference vectors that prioritize one of the criteria that corresponds to a lexicographical order in exact methods. The choice of the preference vector is related to the nature of the final objective and the initial solution. In other words, for an initial solution of $RQCH$, the preference vector that defines the weighting function of SA is then $(1,|N| * T)$, thus prioritizing machine qualifications which is the main objective of QCH.
- The three classic parameters of an SA procedure are initialized. These parameters are problem dependent and we provide below the best found combination of values obtained after several trials:
  - T_o: Initial temperature (or alternatively an initial acceptance probability P_o). The initial value of the temperature is significant for the performance of the SA algorithm. If it is set too high, the

algorithm may spend a long time on poor solutions. If it is set too low, then the algorithm may not perform better than any of the constructive heuristics. We considered an initial temperature $T_o = 20,000$.

- $\alpha < 1$: Cooling factor.The cooling factor used in our experiments is set to 0.95.
- t: Length of temperature step in the cooling schedule. The cooling schedule is the way the temperature is reduced throughout the algorithm. We considered a cooling schedule defined by $T(t) = \alpha * T(t-1)$, which rapidly reduces the temperature and most of the running time is spent at low temperatures. However, linear cooling schemes assign the same importance to all temperatures.

- A stopping criterion is fixed:
  - N_{stop}: Maximum number of iterations without improvement. The SA algorithm stops in our adapted version either because the maximum number of iterations or the running time of 600 seconds is attained. Parameters such as initial temperature and cooling schedule are taken into consideration for convergence to ensure that the temperature is sufficiently low when the stopping rule is satisfied. Experiments were conducted with different stopping rules, and $10,000$ temperature changes are used as the stopping criterion.

- A neighborhood $S(s)$ of feasible solutions in the vicinity of a solution s is defined. This definition is problem dependent. In both problems, different types of neighborhoods are studied. First, it is important to mention that, in our problems, the solution space consists at maximum of all permutations of jobs, i.e. $n!$. SA tries to minimize, for PTC, or maximize, for $PEHF$, the objective function with a predefined preference vector by examining the solution space using moves from permutation to permutation. We call a neighbor of s, a permutation that is reachable by one move. The way neighbors are generated impacts the efficiency of SA. Based on our preliminary experiments on neighborhood structures, two different ways of generating a neighbor were selected for both problems:
  - **Intra-change insertion** of jobs means that a job on a given machine at the jth position is selected and inserted before another job at the ith position on the same machine. Intra-change insertion

also covers *Intra-change Swapping*, where two jobs are randomly selected and swapped on the same machine. Intra-change insertion is found to be more flexible since any swap move may be achieved by two insertion but the inverse is not true. However, the difficulty in our neighborhood generation lies in machine qualifications because some intra-change insertions may lead to additional qualification losses and to a non feasible sequence of jobs on a given machine. Thus, each time an intra-change insertion is tried, the obtained sequence is tested for feasibility since an earlier loss in machine qualifications for a certain job family will for sure inhibit any job of this family to be scheduled after the time instant the machine is disqualified.

  - **Inter-change insertion** corresponds to job positions that are swapped between different machines. For any selection of a job from a machine and its insertion on another machine, the sequence on both machines should be checked for feasibility. Moreover, in an inter-change insertion, the problem of machines with no jobs after a move must be considered. In some situations, there may still only be one job on a machine and an insertion of this job on another machine will lead to a machine with no jobs, thus the machine is not used at all and the number of machines in this case is decreased by 1. To overcome this difficulty, we check, each time a move is performed, whether the machine has strictly more than one job. This guarantees that there is no machine idle on the whole time horizon.

Mathematical representation

The basic elements of simulated annealing are:

- A finite set of solutions S.
- A real-valued cost function f defined on S. Let $S^* \subset S$ be the set of global minima of the function f(e.g. PTC), assumed to be a proper subset of S.
- For each $s \in S$, a set $S(s) \subset S - \{s\}$, called the set of neighbors of s.
- For every s, a collection of positive coefficients $p_{s,s'}$, $s' \in S(s)$, such that $\sum_{s' \in S(s)} p_{s,s'} = 1$. It is assumed that $s' \in S(s) \Leftrightarrow s \in S(s')$. $p_{s,s'}$ is the value of the probability that any particular neighbor $s' \in S(s)$ is selected at random.

- A non-increasing function $T :\to (0, \infty)$, called the *cooling schedule* and $T(t)$ is called the *temperature* at time t.
- An initial solution s_o (a solution is usually associated with a solution *state*).

A sketch of SA, adapted from (Eglese (1990)), is given below.

- Select an initial solution $s \in S$,
- Select the temperature change counter t,
- Select a temperature cooling schedule, $T(t)$,
- Select an initial temperature $T(t_o)$,
- Select a repetition schedule M_t that defines the number of iterations executed at each temperature $T(t)$,
- *Repeat*
- Set the counter $Counter = 0$,
  - *Repeat*
  - Generate a solution $s' \in S(s)$,
  - Calculate $\delta_{s,s'} = Cost(s') - Cost(s)$,
    - If $\delta_{s,s'} \leq 0$, then $s = s'$,
    - Else, $s = s'$ with probability $\exp(-\delta_{s,s'}/T(t))$,
  - Counter = Counter + 1,
  - *Until* Counter = M_t,
- t = t + 1,
- *Until* stopping criterion is met (N_{stop}).

5.5 Numerical experiments on *PTC*

In this section, the results of numerical experiments on PTC obtained for LH, SCH, QCH, and their derivatives, i.e. Recursive LH (RLH), Recursive SCH ($RSCH$), Recursive QCH ($RQCH$), Simulated Annealing based on LH solution ($SABLH$), Simulated Annealing based on SCH solution ($SABSCH$) and Simulated Annealing based on QCH solution ($SABQCH$), are shown and analyzed.

5.5.1 Results on List heuristics (LH, RLH, $SABLH$)

The results of LH and its derivatives (RLH and $SABLH$) are shown in Table 5.2. The results of $IP3$ for a preference vector (1,$|N| * T$) are recalled for the sake of comparison. Prioritizing machine qualifications is considered for the objective functions used in $SABLH$ and $IP3$, since we believe that it is the main criterion to check when comparing LH which depends on scheduling the jobs of different job families based on the shortest threshold first rule. Moreover, the originality of PTC lies in adding machine qualifications to a classical scheduling criterion. Nevertheless, results for the sum of completion times and the losses in machine qualifications are provided. The same selected instance types used in Chapter 4 are being used in this chapter and comments are done on representative instance types for the sake of clarity.

Table 5.2: List Heuristic (LH) - ($\alpha = 1$, $\beta = |N| * T$)

	$\sum_f C_f$				$\sum_f \sum_m Y_f^m$				CPU Time			
No.	LH	RLH	SABLH	IP3	LH	RLH	SABLH	IP3	LH	RLH	SABLH	IP3
1	255	255	255	255	0.4	0.2	0.1	0.0	0	0	0	0.8
2	245	245	245	284	0.9	0.8	0.6	0.0	0	0	0	1.8
3	669	669	669	669	0.9	0.7	0.7	0.5	0	0	0	3.5
4	290	246	246	231	2.8	2.7	2.7	1.7	0	0	0	17
5	323	297	297	278	2.9	2.8	2.8	2.6	0	0	0	600
6	-	-	-	351	-	-	-	0.6	-	-	-	600
7	153	136	136	133	2.8	2.1	1.9	0.8	0	0	0	2.8
8	91	80	81	78	4.8	4.3	3.1	1.9	0	0	0	4.9
9	129	116	116	116	1.2	1.0	0.8	0.0	0	0	0	0.3
10	52	46	46	46	2.1	1.7	1.6	0.0	0	0	0	0.1
11	123	96	96	96	1.1	0.9	0.4	0.0	0	0	0	0.5
12	80	67	67	67	2.2	1.8	1.6	0.1	0	0	0	0.5
13	34	34	34	34	0.3	0.2	0.1	0.0	0	0	0	0.1
14	391	358	394	393	3.8	3.5	2.5	0.6	0	0	0	600
15	359	355	331	313	6.9	6.4	0.6	5.8	0	0	0	227.5
16	404	404	404	404	0.4	0.4	0.3	0.1	0	0	0	2.8
17	167	167	167	163	2.1	1.3	0.9	0.1	0	0	0	1.9
18	473	438	438	407	2.8	2.4	2.0	0.1	0	0	0	30.1
19	340	313	313	329	7.7	6.1	5.7	4.5	0	2	0	600
20	1182	1182	1182	1163	0.4	0.4	0.3	0.2	0	0	0	3.4
21	566	559	559	707	1.9	1.7	1.5	0.1	0	0	0	600
22	574	618	535	512	2.2	1.3	0.7	0.2	0	0	0	230.6
23	690	687	591	674	8.2	6.3	5.8	5.6	0	2	0	600
24	660	627	627	753	2.9	2.7	2.5	0.9	0	0	0	600
25	497	498	498	460	6.2	4.9	4.5	0.7	0	0	0	600
26	1148	1148	1148	1108	0.4	0.3	0.3	0.3	0	0	0	1.6
27	2194	2010	2010	2064	1.2	0.8	0.6	0.2	0	0	0	29.7
28	1671	1671	1692	1135	0.8	0.7	0.3	0.1	0	0	4	18.7
29	2486	2335	1725	1734	5.3	4.2	4.1	3.7	0	0	3	600
30	1746	1746	1639	1355	0.4	0.4	0.3	0.1	0	0	2	126.8
31	4133	4133	2654	2394	0.4	0.4	0.2	0.1	0	0	51	1.2
32	3685	3243	3243	3778	1.1	0.6	0.6	0.2	0	0	0	600
33	2765	2649	2650	2842	2.9	2.5	1.7	0.2	0	0	166	600
34	3452	3158	2710	2966	1.2	0.9	0.7	0.1	0	0	9	600
35	2440	2440	2232	1926	0.4	0.4	0.2	0.1	0	0	9	44
36	1861	1861	1696	1329	0.4	0.4	0.2	0.2	0	0	6	39.7
37	1615	1505	1505	1483	2.2	1.9	1.8	0.5	0	0	2	600
38	1137	1104	1018	1146	5.8	4.3	3.7	0.2	0	0	0	600
39	3823	3823	3628	3129	1.2	1.2	0.8	0.2	0	0	22	600
40	3064	2634	2696	3421	6.8	6.6	5.4	7.8	0	0	21	600
41	1304	1227	1229	1195	4.1	3.2	2.3	0.3	0	0	6	600
42	878	850	909	998	3.9	3.5	3.1	0.5	0	0	179	600
43	3876	3679	3697	3820	2.2	2.0	1.2	0.3	0	0	223	600
44	3329	3471	3068	3674	4.8	4.3	3.1	1.6	0	0	319	600
45	3001	2931	2931	2603	1.2	1.0	0.7	0.2	0	0	10	59.6
46	3715	3477	3421	3945	4.4	4.3	4.1	3.8	0	0	35	600
47	3650	3650	3650	4030	3.4	3.2	3.2	3.9	0	0	2	600
48	2963	2874	2660	2727	2.1	1.8	0.9	0.1	0	0	305	600
49	2876	2868	2735	-	3.4	3.2	3.0	-	0	0	37	-
50	3150	2948	2876	-	4.4	4.0	2.5	-	0	0	48	-

The percentage variation on the sum of completion times with respect to the values obtained by *SABLH* is shown in Figure 5.3. The curves correspond to *LH*, *RLH*, *SABLH* and *IP*3. The preference vector for both *SABLH* and *IP*3 is (1,β_o) that corresponds to prioritizing machine qualifications over the sum of completion times. *SABLH* dominates *LH* and *RLH* in most cases except for Instances 14 and 19. This can be explained because the objective function (scoring) of *SA* prioritizes machine qualifications. Also, it is important to mention that *SA* is applied with an initial solution obtained by *RLH*. On the other hand, *IP*3 dominates *SABLH* for some Instance types (22, 25, 39 and 41), and is dominated for Instance types 19, 21, 38 and 44. However, the values obtained by *IP*3 do not correspond to optimal solutions.

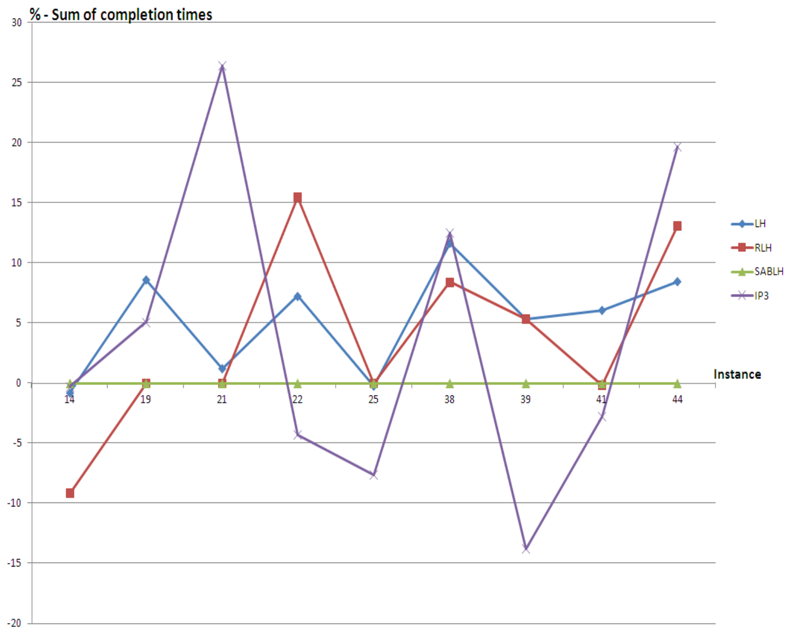


Figure 5.3: *LH*, *RLH*, *SABLH* and *IP*3 - ($\alpha = 1$, $\beta = \beta_o = |N| * T$) - Sum of completion times

Figure 5.4 presents the values of the sum of losses in machine qualifications of *LH*, *RLH*, *SABLH* and *IP*3 with (1,β_o) as a preference vector for *SABLH* and *IP*3. Regarding machine qualifications, *SABLH* is found to dominate *LH* and *RLH*. *RLH* dominates *LH* and this is because it is constructed to minimize the loss in machine qualifications. Moreover, this performance of *RLH* is due to the pre-disqualification of a machine after having a feasible solution from a certain heuristic (*LH* in this case). This information on which machine to disqualify helps in directing the heuristic to search for a better solution in terms of machine qualifications.

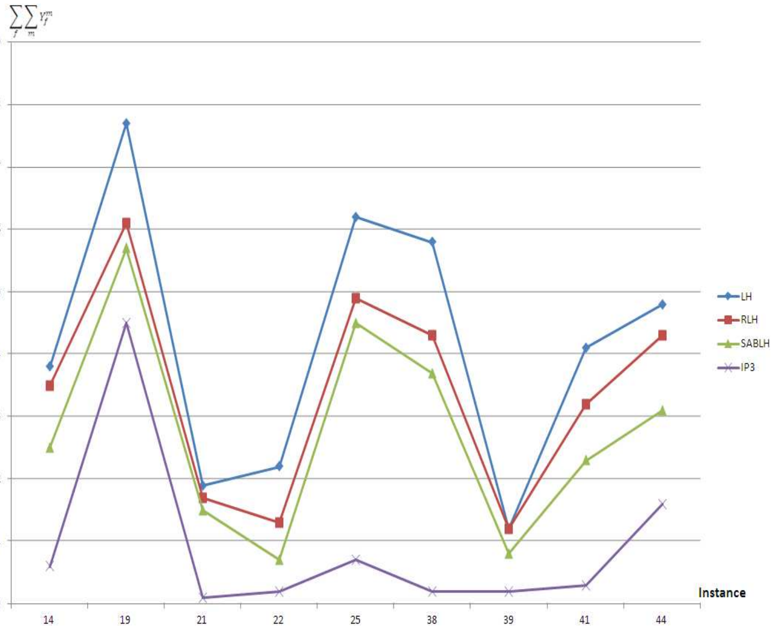


Figure 5.4: LH, RLH, $SABLH$ and $IP3$ - ($\alpha = 1$, $\beta = \beta_o = |N| * T$) - Sum of losses in machine qualifications

LH and RLH solve the selected instance types almost instantaneously. However, $SABLH$ spends more time depending on its input parameters and stopping criteria (initial temperature, number of iterations, etc). Alternatively, Model $IP3$ spends more time than LH and its derivatives as shown in Figure 5.5.

5.5.2 Results on Scheduling-centric heuristics (SCH, $RSCH$, $SABSCH$)

The results of SCH and its derivatives ($RSCH$ and $SABSCH$) are shown in Table 5.3. The results of $IP3$ for a preference vector (1,1) are recalled for the sake of comparison. Prioritizing the sum of completion times is considered for the objective functions used in $SABSCH$ and $IP3$, since it seems important to check the performance of SCH and its derivatives based on this criterion. SCH is designed to first minimize the sum of completion times by scheduling jobs of same job families as far as it is possible without violating a time threshold corresponding to another job family. Results for the sum of

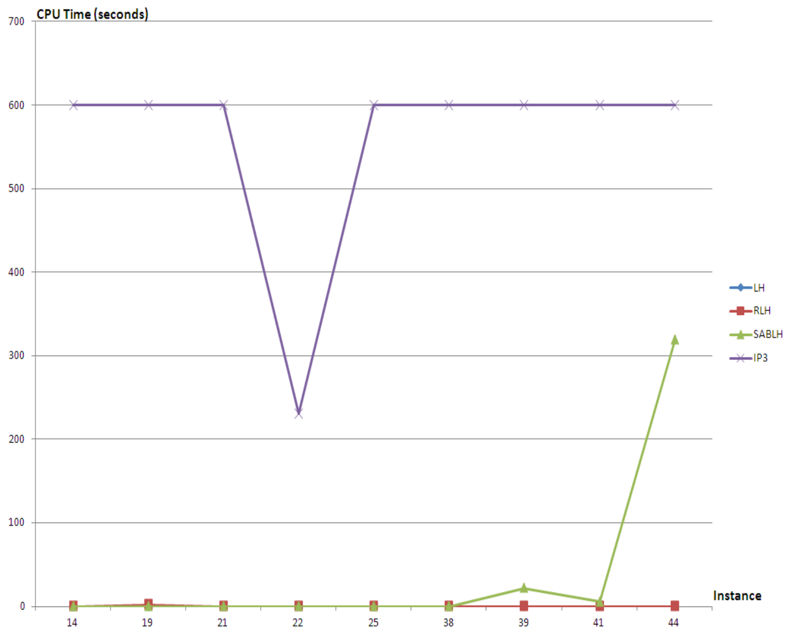


Figure 5.5: LH, RLH, $SABLH$ and $IP3$ - ($\alpha = 1$, $\beta = \beta_o = |N| * T$) - CPU times

completion times and losses in machine qualifications are provided for SCH, $RSCH$, $SABSCH$ and $IP3$. The same instance types selected in Chapter **??** are also being used, comments are done on representative instance types and a graphical interpretation is provided.

In Figure 5.6, the percentage of increase in the sum of completion times for SCH and its derivatives, with respect to $IP3$ with a preference vector (1,1), is presented. As the figure shows, $SABSCH$ of a preference vector (1,1) dominates SCH and $RSCH$. Exact solutions are attained for some instance types (e.g. 9 and 21). However, for $SABSCH$ in Instance type 39, the percentage increase is around 22% which is related to a corresponding *gain* in machine qualifications (the same instance in Figure 5.7), where $IP3$ loses on average 2.2 machine qualifications.

The sum of losses in machine qualifications is presented for SCH, $RSCH$, $SABSCH$ and $IP3$ in Figure 5.7. The figure shows that $RSCH$ dominates SCH, $SABSCH$ and $IP3$. This is due once again to the nature of the recursive algorithm that tends to accept solutions with minimized losses in machine qualifications. It is important to recall that both $SABSCH$ and $IP3$ have (1,1) as a preference vector, hence the sum of completion time is

Table 5.3: Scheduling-Centric Heuristic (SCH) - ($\alpha = 1$, $\beta = 1$)

	$\sum_{f \in F} C_f$				$\sum_f \sum_m Y_f^m$				CPU Time			
$No.$	SCH	RSCH	SABSCH	IP3	SCH	RSCH	SABSCH	IP3	SCH	RSCH	SABSCH	IP3
1	255	255	255	255	0.4	0.2	0.2	0.0	0	0	0	0.4
2	245	245	245	245	0.9	0.9	0.9	0.9	0	0	0	0.4
3	669	669	669	564	0.9	0.7	0.7	1.6	0	0	0	0.8
4	263	246	246	211	2.8	2.6	2.6	3.1	0	0	0	1.7
5	323	323	323	278	2.9	2.5	2.5	3.1	0	0	0	6.4
6	-	-	-	351	-	-	-	1.2	-	-	-	16.8
7	153	130	130	124	2.8	2.5	2.6	2.1	0	0	0	4
8	96	82	78	76	4.7	3.5	4.6	3.3	0	0	0	2
9	129	116	116	116	1.2	0.6	0.8	0.4	0	0	0	0.3
10	52	46	46	46	1.9	1.5	1.7	0.4	0	0	0	0.1
11	123	96	96	96	1.0	0.7	0.8	0.4	0	0	0	0.5
12	80	67	67	67	2.1	1.8	1.9	0.4	0	0	0	0.5
13	34	34	34	34	0.3	0.3	0.3	0.3	0	0	0	0.1
14	393	358	358	346	3.7	3.5	3.6	3.3	0	0	0	13
15	335	297	297	285	6.7	6.5	6.7	6.6	0	0	0	3.6
16	404	404	404	404	0.3	0.3	0.3	0.3	0	0	0	3.2
17	167	167	167	157	2.1	1.0	1.1	1.2	0	0	0	2.2
18	473	438	438	399	2.8	1.6	1.7	1.4	0	0	0	27.4
19	340	297	297	297	7.7	7.6	7.6	7.2	0	0	0	47.8
20	1182	1182	1182	1163	0.4	0.3	0.3	0.0	0	0	0	4.2
21	566	559	559	559	1.9	1.8	1.8	2.4	0	0	0	7.7
22	550	550	524	483	1.8	1.5	1.6	2.7	0	0	3	10.4
23	-	-	-	538	-	-	-	7.3	-	-	-	95.4
24	673	637	637	621	2.8	2.5	2.5	3.4	0	0	0	31.2
25	496	481	477	427	6.8	5.4	6.7	6.1	0	0	0	46.5
26	1148	1148	1148	1108	0.4	0.3	0.4	0.4	0	0	0	1.5
27	2194	2010	2010	2010	1.2	1.2	1.2	1.2	0	0	0	10.7
28	1671	1671	1461	1117	0.7	0.3	0.6	1.1	0	0	32	7.3
29	-	-	-	1608	-	-	-	6.4	-	-	-	243.6
30	1746	1746	1659	1355	0.4	0.3	0.4	0.4	0	0	9	102.7
31	4133	4133	2622	2394	0.4	0.3	0.4	0.4	0	0	119	1.3
32	3685	3243	3243	3243	1.1	1.0	1.2	2.2	0	0	0	7.2
33	2765	2649	2649	2621	2.9	2.6	2.6	2.1	0	0	24	108.6
34	3452	3158	2798	2710	1.2	1.0	1.1	2.3	0	0	56	40.4
35	2440	2440	2224	1926	0.4	0.3	0.4	0.4	0	0	51	43.8
36	1913	1913	1675	1329	0.3	0.3	0.3	0.4	0	0	92	31.9
37	1615	1505	1505	1294	2.2	2.1	2.1	3.4	0	0	0	306.5
38	1128	1073	1004	962	5.7	4.2	4.4	5.4	0	0	1	78.8
39	3823	3823	3629	2992	1.2	1.1	1.2	3.3	0	0	82	137.4
40	3074	2535	2535	2504	7.2	6.3	6.3	9.1	0	0	12	345.1
41	1280	1210	1208	1158	3.8	3.5	3.6	4.3	0	0	23	106.7
42	873	838	838	834	3.7	3.9	3.9	6.2	0	0	126	47
43	3800	3657	3638	3626	2.9	1.8	2.1	2.4	0	0	186	33.2
44	3189	3129	2873	2711	4.9	4.5	4.7	5.3	0	0	189	213.7
45	3001	2931	2931	2603	1.2	0.6	0.6	0.4	0	0	0	58.1
46	3698	3452	3452	4060	4.2	3.8	3.8	4.3	0	0	109	600
47	3650	3588	3588	3429	3.4	2.9	2.9	3.1	0	0	15	600
48	2963	2874	2486	2389	2.1	1.6	1.7	2.3	0	0	372	600
49	2870	2854	2720	-	3.3	3.0	3.1	-	0	0	62	-
50	2952	2762	2653	-	4.2	3.7	3.9	-	0	0	56	-

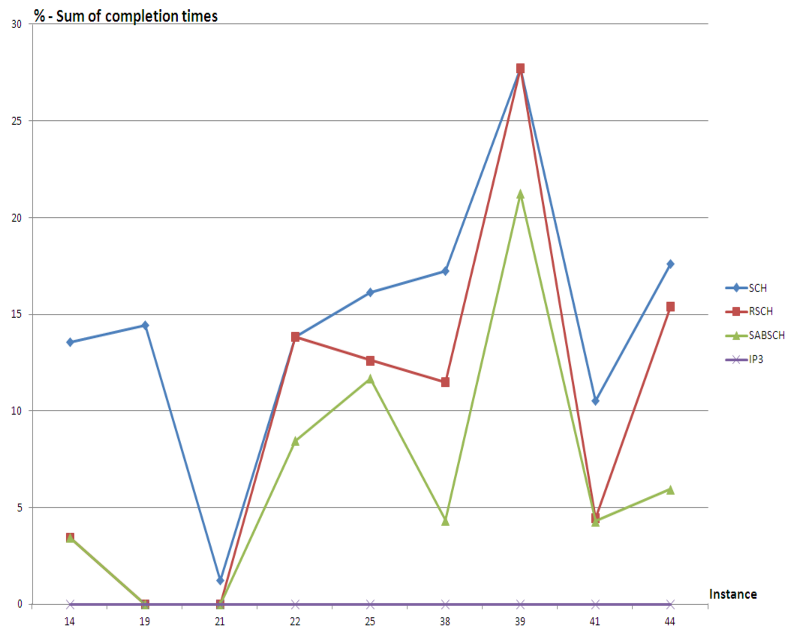


Figure 5.6: SCH, $RSCH$, $SABSCH$ and $IP3$ - ($\alpha = 1$, $\beta = 1$) - Sum of completion times

prioritized.

Figure 5.8 shows the CPU times of SCH, $RSCH$, $SABSCH$ and $IP3$ with a preference vector (1,1). When comparing the obtained CPU times with the ones obtained for (1,β_o), it can be observed that $IP3$ solves all the selected instance types to optimality for a preference vector (1,1) which is not the case for a preference vector (1,β_o). This illustrates the complexity added to PTC whenever it is decided to prioritize and keep on qualified machines. Hence, time constraints play an important role in the complexity of PTC.

5.5.3 Results on Qualification-centric heuristics (QCH, $RQCH$, $SABACH$)

The results of QCH and its derivatives ($RQCH$ and $SABQCH$) are shown in Table 5.4. The results of $IP3$ for a preference vector (1,$|N| * T$) are recalled for the sake of comparison. Prioritizing machine qualifications is considered for the objective functions used in $SABQCH$ and $IP3$, since it seems important to check the performance of QCH regarding machine qualifications. Recall that QCH is designed to first minimize the total sum of losses in

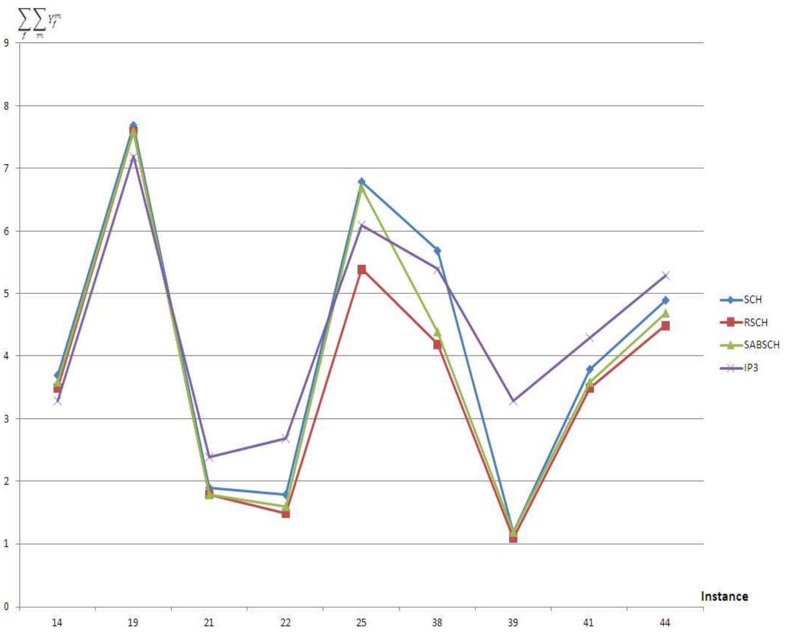


Figure 5.7: SCH, $RSCH$, $SABSCH$ and $IP3$ - ($\alpha = 1$, $\beta = 1$) - Sum of losses in machine qualifications

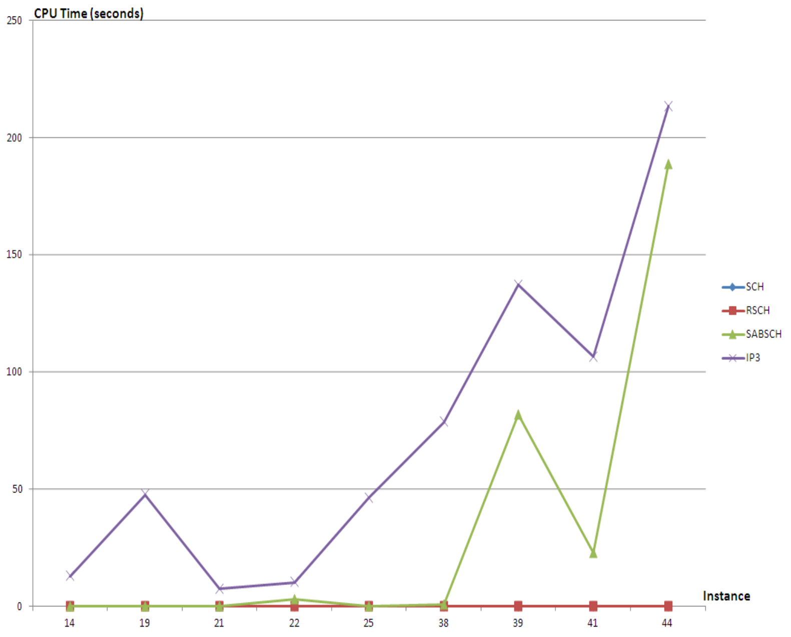


Figure 5.8: SCH, $RSCH$, $SABSCH$ and $IP3$ - ($\alpha = 1$, $\beta = 1$) - CPU times

machine qualifications by scheduling jobs according to their updated family time threshold. Results for the sum of completion times and losses in machine qualifications are provided for QCH, $RQCH$, $SABQCH$ and $IP3$.

Table 5.4: Qualification-Centric Heuristic (QCH) - ($\alpha = 1$, $\beta = |N| * T$)

	$\sum_{f \in F} C_f$				$\sum_f \sum_m Y_f^m$				CPU Time			
$No.$	QCH	RQCH	SABQCH	IP3	QCH	RQCH	SABQCH	IP3	QCH	RQCH	SABQCH	IP3
1	255	255	255	255	0.3	0.1	0.0	0.0	0	0	0	0.8
2	303	303	303	284	0.4	0.2	0.0	0.0	0	0	0	1.8
3	669	669	669	669	0.8	0.8	0.6	0.5	0	0	0	3.5
4	290	246	246	231	2.7	2.5	2.5	1.7	0	0	0	17
5	323	297	297	278	2.8	2.7	2.6	2.6	0	0	0	600
6	-	-	-	351	-	-	-	0.6	-	-	-	600
7	189	136	136	133	1.7	1.5	1.5	0.8	0	0	0	2.8
8	99	99	100	78	2.8	2.5	1.9	1.9	0	0	0	4.9
9	165	165	165	116	0.4	0.3	0.0	0.0	0	0	0	0.3
10	92	92	92	46	0.3	0.1	0.1	0.0	0	0	0	0.1
11	141	141	141	96	0.4	0.2	0.0	0.0	0	0	0	0.5
12	80	67	67	67	1.8	1.6	1.6	0.1	0	0	0	0.5
13	34	34	34	34	0.3	0.3	0.1	0.0	0	0	0	0.1
14	436	436	452	393	2.8	2.7	1.5	0.6	0	0	0	600
15	335	350	349	313	6.7	6.1	5.8	5.8	0	0	0	227.5
16	404	404	404	404	0.3	0.1	0.1	0.1	0	0	0	2.8
17	167	167	167	163	1.8	1.3	0.5	0.1	0	0	0	1.9
18	503	439	438	407	1.9	1.7	1.6	0.1	0	0	0	30.1
19	378	378	313	329	5.8	5.7	5.5	4.5	0	0	0	600
20	1239	1239	1182	1163	0.3	0.3	0.2	0.2	0	0	0	3.4
21	745	745	745	707	0.9	0.7	0.5	0.1	0	0	0	600
22	544	535	532	512	0.8	0.6	0.6	0.2	0	0	0	230.6
23	707	687	609	674	7.7	6.3	6.1	5.6	0	0	2	600
24	859	859	844	753	1.8	1.8	1.6	0.9	0	0	0	600
25	679	679	667	460	1.9	1.7	1.5	0.7	0	0	0	600
26	1572	1572	1148	1108	0.4	0.4	0.3	0.3	0	0	0	1.6
27	2429	2010	2010	2064	0.8	0.7	0.5	0.2	0	0	0	29.7
28	1210	1210	1210	1135	0.4	0.2	0.1	0.1	0	0	3	18.7
29	2924	2924	1732	1734	4.1	3.9	3.7	3.7	0	0	19	600
30	2007	2007	1651	1355	0.3	0.3	0.1	0.1	0	0	8	126.8
31	4133	4133	2708	2394	0.4	0.4	0.1	0.1	0	0	111	1.2
32	3685	3243	3243	3778	1.0	0.8	0.6	0.2	0	0	0	600
33	2947	2649	2672	2842	2.7	2.5	1.7	0.2	0	0	95	600
34	3075	3075	3075	2966	0.8	0.6	0.5	0.1	0	0	0	600
35	2440	2440	2440	1926	0.4	0.1	0.1	0.1	0	0	0	44
36	1851	1851	1634	1329	0.3	0.2	0.2	0.2	0	0	36	39.7
37	1957	1505	1505	1483	1.8	1.7	1.5	0.5	0	0	0	600
38	1301	1019	1019	1146	3.7	3.6	3.5	0.2	0	0	0	600
39	5200	3823	3634	3129	0.8	0.7	0.6	0.2	0	0	75	600
40	3935	3935	3980	3421	4.4	4.2	3.3	7.8	0	0	214	600
41	1568	1568	1561	1195	1.9	1.6	0.3	0.3	0	0	4	600
42	1055	1055	1055	998	2.8	2.7	2.5	0.5	0	0	104	600
43	4593	3662	3712	3820	1.7	1.5	0.6	0.3	0	0	186	600
44	3591	3455	3082	3674	2.8	2.6	1.7	1.6	0	0	215	600
45	3223	2931	2931	2603	0.9	0.7	0.5	0.2	0	0	0	59.6
46	4249	3463	3463	3945	4.2	4.0	3.8	3.8	0	0	27	600
47	5634	5634	5730	4030	2.3	2.3	1.3	3.9	0	0	1	600
48	3427	2874	2506	2727	1.7	1.5	0.6	0.1	0	0	309	600
49	3074	2986	2888	-	2.2	2.1	1.9	-	0	0	27	-
50	3582	3373	3257	-	3.1	2.7	2.3	-	0	0	32	-

The percentage variation in the sum of completion times for QCH, $SABQCH$ and $IP3$ with respect to $RQCH$ is shown in Figure 5.9. The results obtained with $IP3$ and $SABQCH$ correspond to a preference vector (1,NT) where machine qualification is prioritized. As the figure shows, $IP3$ dominates for Instance types 14, 21, 22, 25, 39 and 41. Alternatively, $SABQCH$ dominates for Instance types 19, 38 and 44. On the other hand, it is found that the difference in losses in machine qualifications between $IP3$ and $SABQCH$

is maximized for the same instances for which $SABQCH$ dominates $IP3$ as shown in Figure 5.10. Further, in this figure, $IP3$ dominates QCH and its derivatives. Note also that $SABQCH$ almost attains the best solutions found by $IP3$ for Instance types 41 and 44. Moreover, $RQCH$ dominates QCH regarding the sum of losses in machine qualifications as Figure 5.10 demonstrates.

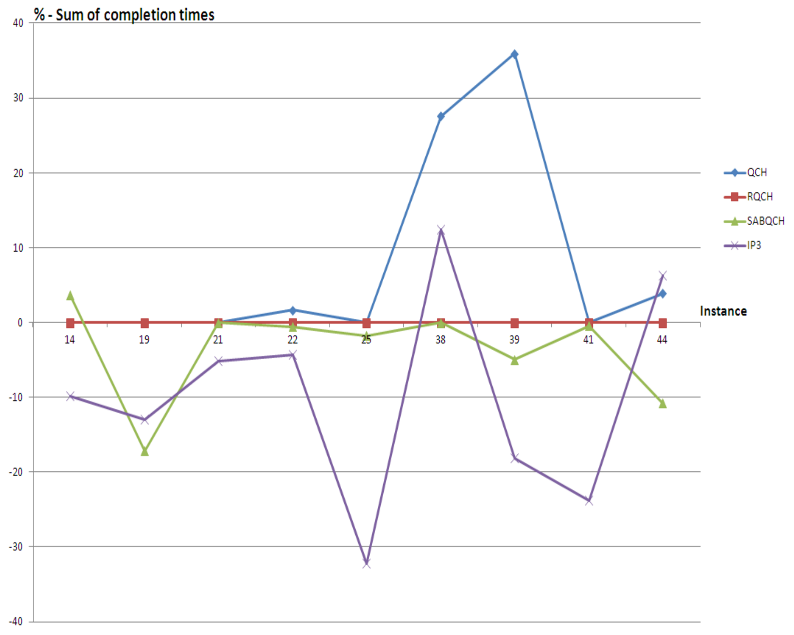


Figure 5.9: QCH, $RQCH$, $SABQCH$ and $IP3$ - ($\alpha = 1$, $\beta = \beta_o = |N| * T$) - Sum of completion times

CPU times for $IP3$ attain the time limit for 8/9 of the selected instance types. This can be explained when knowing that prioritizing machine qualifications forces the standard solver to respect time constraints. QCH and $RQCH$ solves the instances instantaneously. For $SABQCH$, considerable CPU times are found for large instances (e.g. Instance types 39 and 44). These values are also affected by the initial settings of the SA parameters such as the initial temperature, cooling factor, number of iterations, etc.

5.6 Numerical experiments on $PEHF$

The instances used for PTC are also used for $PEHF$. However, a new matrix with the values of yield for each family to machine assignment is now

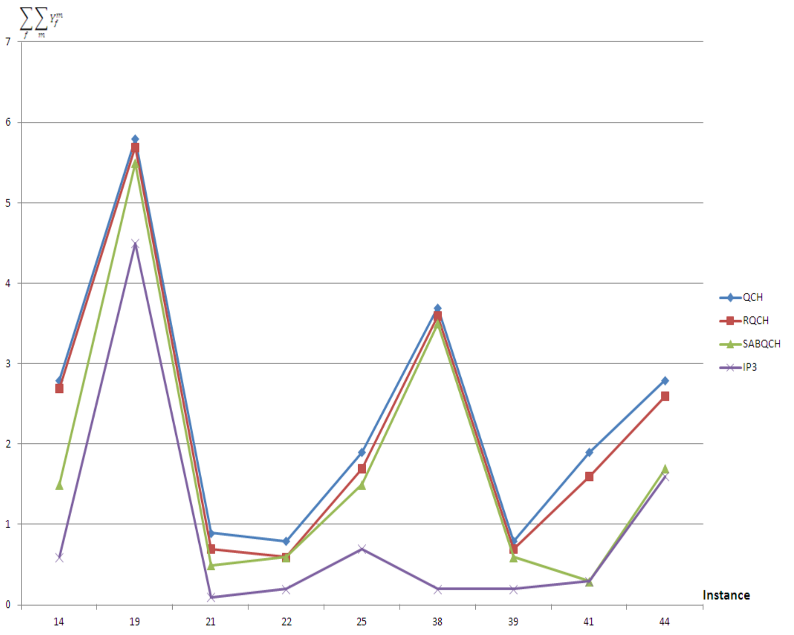


Figure 5.10: QCH, $RQCH$, $SABQCH$ and $IP3$ - ($\alpha = 1$, $\beta = \beta_o = |N| * T$) - Sum of losses in machine qualifications

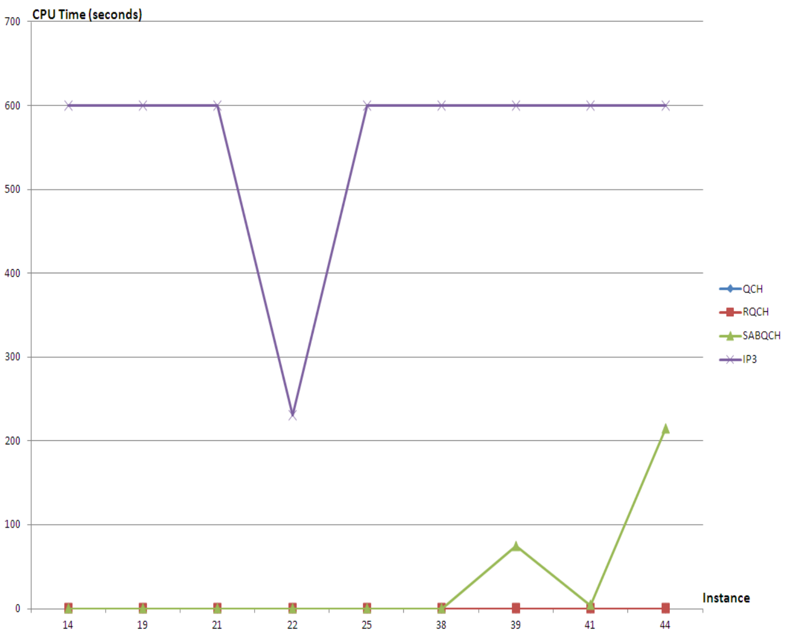


Figure 5.11: QCH, $RQCH$, $SABQCH$ and $IP3$ - ($\alpha = 1$, $\beta = \beta_o = |N| * T$) - CPU times

necessary. Moreover, the values of the yield in this matrix are generated such that, whenever a machine m is *healthier* than a machine m', then m has a greater yield than m' for all job families. Results for SCH, QCH and YCH are provided and compared. Also, results for YCH, $RYCH$ and $SABYCH$ are also provided for a preference vector (1,1,γ_o), that prioritizes the expected yield over the sum of completion times and machine qualifications.

5.6.1 SCH, QCH and YCH

Table 5.5 presents the sum of completion times, the sum of losses in machine qualifications and the sum of expected yield obtained for SCH, QCH and YCH. Simulations showed that some instance types are not feasible for one constructive heuristic but feasible for another. For example, Instance type 6 is feasible for YCH but not for SCH and QCH, whereas Instance type 14 is feasible for SCH and QCH but not for YCH. Instance types 23 and 29 are other examples. YCH has the best solution on the sum of expected yield in 41 out of 47 solved instances compared to SCH and QCH. QCH has the best solutions on the sum of losses in machine qualifications for 45 out of 49 solved instances compared to SCH and YCH. On the total sum of completion times, SCH is found to be the best for 29 out of 47 solved instances. However, YCH is also effective for the sum of completion times (best solutions are found for 26 out of 49 solved instances). This is explained by the tendency of YCH to schedule jobs of the same family on the machine with the best EHF, thus minimizing the number of setups and consequently the sum of completion times. CPU times are omitted since constructive heuristics are very fast.

Table 5.5: Comparison between heuristics ($\sum_{f\in F} C_f$, $\sum_f \sum_m Y_f^m$ and $\sum_{t=1}^{T} \sum_f \sum_m x_{f,t}^m v_{f,t}^m$), best solution in bold

	$\sum_{f\in F} C_f$			$\sum_f \sum_m Y_f^m$			$\sum_{t=1}^{T} \sum_f \sum_m x_{f,t}^m v_{f,t}^m$		
$No.$	SCH	QCH	YCH	SCH	QCH	YCH	SCH	QCH	YCH
1	**255**	**255**	296	0.4	**0.3**	0.4	870	870	**896**
2	**245**	303	284	0.9	**0.4**	1.1	890	**894**	890
3	669	669	**564**	0.9	**0.8**	1.1	924	924	**952**
4	**263**	290	**263**	2.8	**2.7**	3.3	**846**	840	**846**
5	**323**	**323**	-	2.9	**2.8**	-	**844**	**844**	-
6	-	-	**396**	-	-	**2.9**	-	-	**881**
7	153	189	**130**	2.8	**1.7**	3.8	905	895	**907**
8	96	99	**83**	4.7	2.8	**2.3**	926	931	**942**
9	**129**	165	**129**	1.2	**0.4**	1.4	863	**877**	863
10	52	92	**48**	1.9	**0.3**	2.1	**904**	**904**	**904**
11	**123**	141	**123**	1.0	**0.4**	1	913	**921**	913
12	80	80	**67**	2.1	1.8	**0.4**	899	**913**	**913**
13	**34**	**34**	**34**	**0.3**	**0.3**	0.4	**908**	**908**	**908**
14	**393**	436	-	3.7	**2.8**	-	1815	**1833**	-
15	335	335	**326**	6.7	6.7	**3.3**	1785	1785	**1805**
16	**404**	**404**	472	**0.3**	**0.3**	0.4	1794	1794	**1798**
17	**167**	**167**	**167**	2.1	**1.8**	2.3	**1774**	**1774**	**1774**
18	473	503	**457**	2.8	**1.9**	2.3	1760	1740	**1768**
19	**340**	378	367	7.7	**5.8**	7.2	1769	1759	**1777**
20	**1182**	1239	1216	0.4	**0.3**	0.4	2677	**2724**	2709
21	**566**	745	704	1.9	**0.9**	1.8	2632	2677	**2768**
22	550	**544**	581	1.8	**0.8**	2.2	2701	2701	**2705**
23	-	**707**	-	-	**7.7**	-	-	**2696**	-
24	**673**	859	716	2.8	**1.8**	3.2	2762	2765	**2773**
25	**496**	679	498	6.8	**1.9**	5.3	2610	**2647**	2634
26	**1148**	1572	**1148**	**0.4**	**0.4**	**0.4**	**3759**	3663	**3759**
27	**2194**	2429	2435	1.2	**0.8**	1.1	3689	3724	**3736**
28	1671	**1210**	1668	0.7	**0.4**	**0.4**	3542	3536	**3552**
29	-	2924	**2069**	-	**4.1**	4.9	-	3543	**3599**
30	**1746**	2007	**1746**	0.4	**0.3**	**0.3**	**3523**	3510	**3523**
31	**4133**	**4133**	**4133**	0.4	0.4	**0.3**	**4687**	**4687**	**4687**
32	**3685**	**3685**	5429	1.1	**1.0**	1.2	**4154**	**4154**	**4154**
33	**2765**	2947	3259	2.9	**2.7**	3.2	4376	4350	**4382**
34	3452	3075	**2927**	1.2	**0.8**	1.3	4611	4621	**4616**
35	2440	2440	**2126**	**0.4**	**0.4**	**0.4**	4319	4319	**4342**
36	1913	1851	**1850**	**0.3**	**0.3**	1.1	4553	4543	**4560**
37	1615	1957	**1337**	2.2	**1.8**	3.4	4336	4343	**4465**
38	**1128**	1301	1181	5.7	**3.7**	6.2	4429	**4437**	4430
39	**3823**	5200	4432	1.2	**0.8**	1.3	5412	5460	**5502**
40	3074	3935	**2973**	7.2	**4.4**	5.3	5486	5475	**5545**
41	**1280**	1568	1282	3.8	**1.9**	3.4	5362	5284	**5365**
42	**873**	1055	877	3.7	**2.8**	3.8	5316	5341	**5383**
43	**3800**	4593	3837	2.9	**1.7**	2.4	6352	6352	**6446**
44	3189	3591	**3049**	4.9	**2.8**	5.3	6114	6060	**6229**
45	3001	3223	**2877**	1.2	**0.9**	1.4	6376	6222	**6568**
46	3698	4249	**3680**	**4.2**	**4.2**	4.4	6321	6210	**6331**
47	**3650**	5634	3764	3.4	**2.3**	3.9	6168	6227	**6387**
48	2963	3427	**2730**	2.1	**1.7**	2.2	6215	6251	**6287**
49	**2870**	3074	2995	3.3	**2.2**	3.2	6356	6289	**6388**
50	**2952**	3582	3114	4.2	**3.1**	4.1	6402	6398	**6415**

The percentage variation of the sum of completion times are shown in Figure 5.12 for SCH, QCH and YCH relative to the values obtained for SCH. SCH dominates QCH and YCH in terms of minimizing the sum of completion times. It is not the case for Instance types 22 and 44. Again, this is due to the fact that YCH tends to schedule job of the same family on the same machine in order to maximize the resulting yield. In addition, it is possible for QCH to schedule at an early stage jobs of different families except for one, and hence to schedule the remaining jobs of the last family, thus minimizing the number of setups and consequently the sum of completion times.

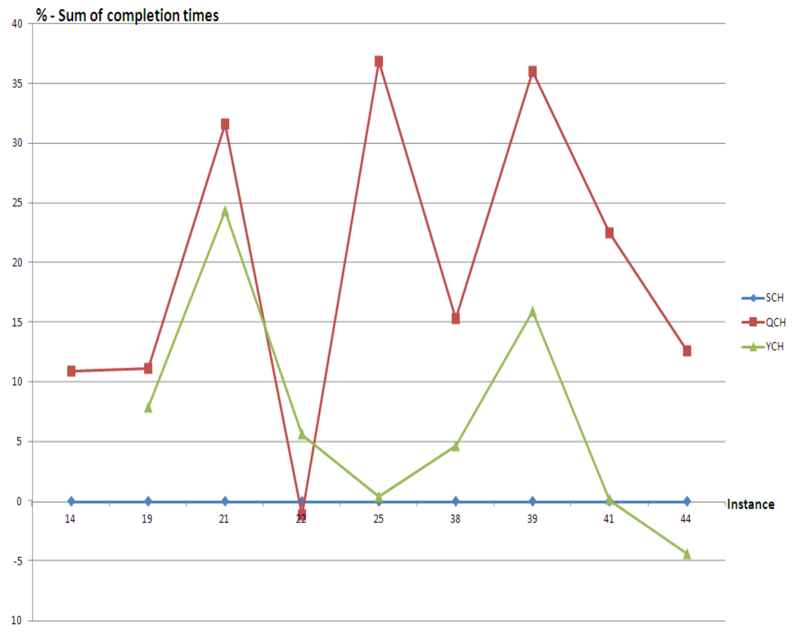


Figure 5.12: SCH, QCH and YCH - Sum of completion times

QCH dominates SCH and YCH on the loss in machine qualifications as shown in Figure 5.13. It is important to note that it is not guaranteed to have a feasible solution for all instances/instance types when applying a certain heuristic. Instance type 14 is an example where YCH has no feasible solution. Moreover, note that SCH and YCH dominate alternatively in terms of machine qualifications, e.g. Instance type 19 (YCH dominates SCH) and Instance type 38 (SCH dominates YCH).

YCH dominates in most cases SCH and QCH on the sum of expected yield. QCH and SCH dominate alternatively (e.g. Instances 19, 41 and 44

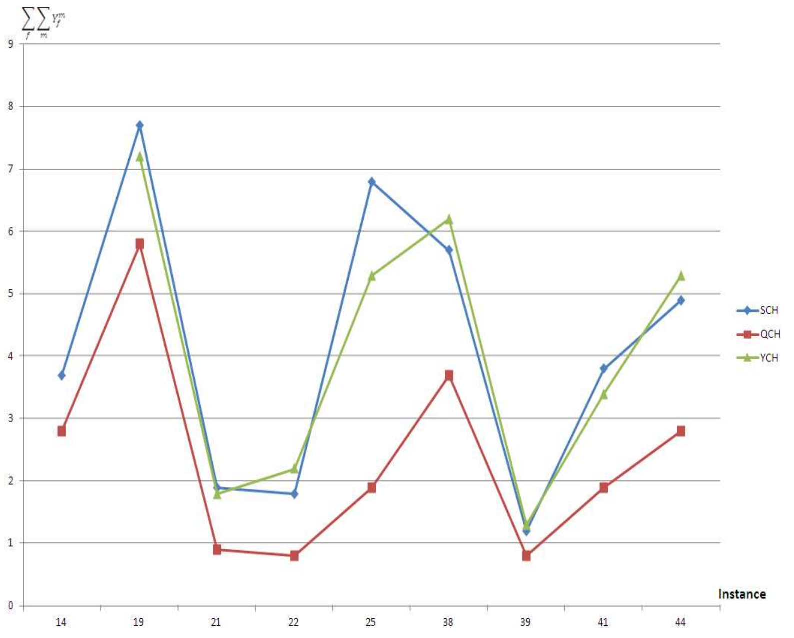


Figure 5.13: SCH, QCH and YCH - Sum of losses in machine qualifications

- SCH dominates QCH, and Instance types 14, 21, 25 and 39, where QCH dominates SCH).

5.6.2 Results on RH and SA

Numerical results obtained for the Recursive Heuristic applied on YCH ($RYCH$) and Simulated Annealing applied on the solution obtained by $RYCH$ ($SABYCH$) with a preference vector (1,1,$|M| * |F| * |N| * T$) are given in Tables 6.a and 6.b. The results are compared to $IP5$ with a preference vector (1,1,$|M| * |F| * |N| * T$) where the sum of expected yield is prioritized on the sum of completion times and machine qualifications. This preference vector is chosen because YCH is designed to maximize the sum of expected yield.

The results in Figure 5.15 through 5.18 correspond to a preference vector (1,1,γ_o) for both $IP5$ and SA, where the expected yield is prioritized on both the sum of completion times and machine qualification. This preference vector is chosen because we wanted to concentrate on the performance of YCH and its derivatives compared to $IP5$ (exact method).

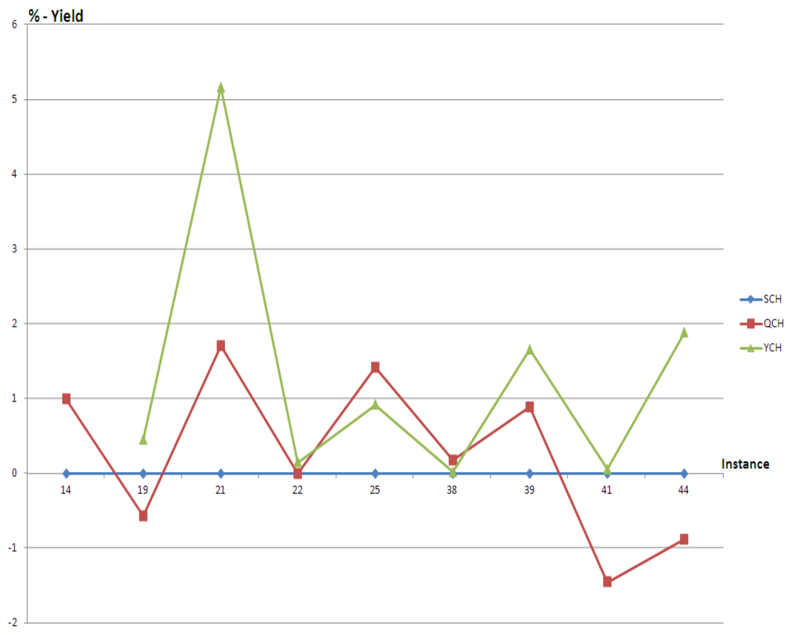


Figure 5.14: SCH, QCH and YCH - Percentage of gain in yield with respect to SCH

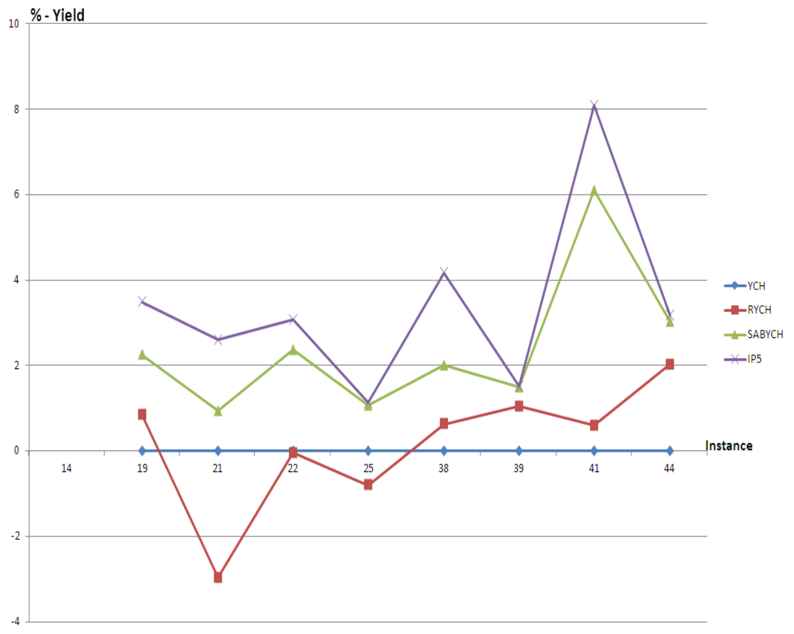


Figure 5.15: YCH, $RYCH$, $SABYCH$ and $IP5$ - ($\alpha = 1$, $\beta = 1$, $\gamma = \gamma_o = |M| * |F| * |N| * T$) - Percentage of gain in yield with respect to YCH

Table 6.a: Yield-Centric Heuristic (YCH) - ($\alpha' = 1$, $\beta' = 1$, $\gamma' = |M| * |F| * |N| * T$), $\sum_{t=1}^{T} \sum_f \sum_m x_{f,t}^m v_{f,t}^m$ and $\sum_f \sum_m Y_f^m$

	$\sum_{t=1}^{T} \sum_f \sum_m x_{f,t}^m v_{f,t}^m$				$\sum_f \sum_m Y_f^m$			
No.	YCH	RYCH	SABYCH	IP5	YCH	RYCH	SABYCH	IP5
1	896	896	904	922	0.4	0.3	0.3	0.1
2	890	890	890	894	1.1	1.0	1.2	0.5
3	952	952	952	952	1.1	1.1	1.3	1.7
4	846	852	853	856	3.3	3.1	3.2	3.6
5	-	-	-	856	-	-	-	3.9
6	881	889	889	896	2.9	2.4	3.1	4.7
7	907	909	933	939	3.8	1.4	1.4	4.8
8	942	946	974	981	2.3	2.3	2.4	9.5
9	863	863	883	915	1.4	1.2	1.4	2.9
10	904	909	916	926	2.1	1.9	2.2	3.6
11	913	913	929	938	1.0	0.9	1.3	4.8
12	913	932	963	982	0.4	0.3	0.4	9.7
13	908	908	940	950	0.4	0.3	0.4	8.8
14	-	-	-	1868	-	-	-	6.5
15	1805	1813	1828	1835	3.3	3.2	3.4	6.9
16	1798	1798	1828	1892	0.4	0.4	0.4	0.4
17	1774	1805	1813	1837	2.3	1.1	1.4	2.6
18	1768	1790	1826	1844	2.3	1.2	1.3	9.7
19	1777	1792	1817	1839	7.2	6.1	7.2	10.7
20	2709	2727	2747	2748	0.4	0.2	0.3	0.4
21	2768	2686	2794	2840	1.8	1.3	2.2	2.9
22	2705	2704	2769	2788	2.2	2.1	2.3	5.8
23	-	-	-	2785	-	-	-	12.6
24	2773	2800	2887	2900	3.2	3.0	3.4	6.9
25	2634	2613	2662	2664	5.3	4.3	4.4	9.5
26	3759	3759	3889	3941	0.4	0.2	0.3	0.3
27	3736	3736	3809	3814	1.1	1.0	1.3	1.7
28	3552	3552	3580	3582	0.4	0.3	0.4	4.6
29	3599	3603	3616	3680	4.9	4.8	5.1	8.6
30	3523	3523	3666	3686	0.3	0.2	0.3	3.8
31	4687	4687	4699	4700	0.3	0.2	0.3	0.3
32	4154	4210	4210	4210	1.2	1.1	1.4	1.7
33	4382	4391	4442	4455	3.2	2.2	2.3	2.5
34	4616	4631	4726	4744	1.3	1.1	1.3	1.8
35	4342	4342	4394	4608	0.4	0.4	0.4	0.3
36	4560	4585	4586	4824	1.1	1.0	1.2	6.7
37	4465	4563	4658	4764	3.4	2.3	3.2	9.9
38	4430	4458	4519	4615	6.2	4.1	4.3	10.6
39	5502	5560	5584	5585	1.3	1.2	1.4	4.8
40	5545	5591	5735	5747	5.3	5.1	5.2	9.7
41	5365	5397	5693	5799	3.4	2.3	3.1	9.9
42	5383	5398	5467	5550	3.8	4.2	4.3	7.9
43	6446	6446	6446	6446	2.4	2.3	2.4	4.8
44	6229	6355	6418	6427	5.3	4.3	4.4	8.6
45	6568	6568	6580	6757	1.4	1.2	1.4	4.5
46	6331	6365	6480	6560	4.4	4.1	4.3	4.7
47	6387	6503	6578	6625	3.9	3.4	4.1	6.7
48	6287	6367	6421	6426	2.2	1.4	2.2	9.6
49	6388	6396	6405	-	3.2	2.9	3.0	-
50	6415	6436	6486	-	4.1	3.8	3.9	-

In Figure 5.15, the variation in percentage of gain/loss in yield of $RYCH$, $SABYCH$ and $IP5$ referenced to the values obtained by YCH are presented. As the figure shows, $SABYCH$ dominates other heuristics and almost attains the exact solutions in some instances (Instance types 25, 39 and 44). However, $RYCH$ is dominated by YCH (Instance types 21 and 25). This is because $RYCH$ accepts a deterioration of one criterion to improve machine qualifications as illustrated in Figure 5.16. Also in this figure, $RYCH$ dominates $SABYCH$ in terms of machine qualifications. This is due to the fact that $SABYCH$ tries to search for solutions that enhance the expected yield compared to machine qualifications (preference vector (1,1,γ_o)). For all selected instance types, $RYCH$ dominates YCH, $SABYCH$ and $IP5$ on machine qualifications.

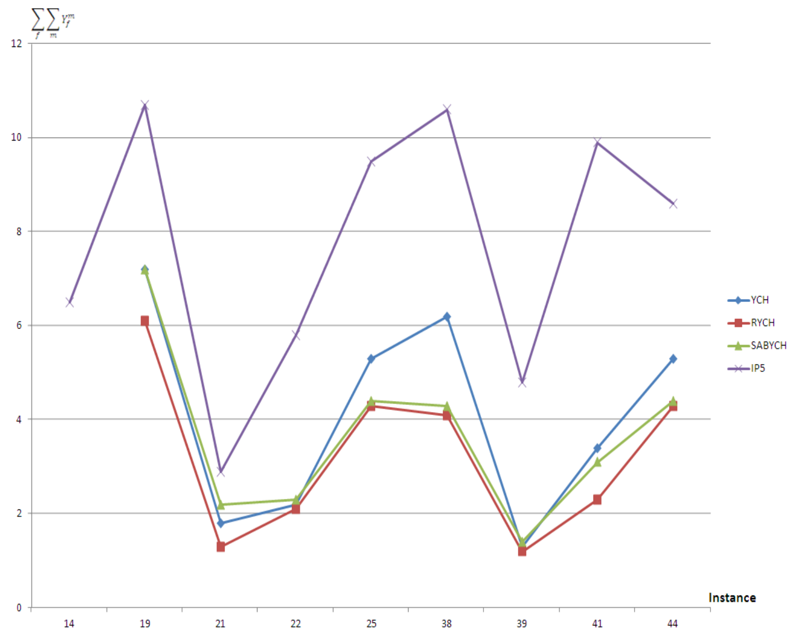


Figure 5.16: YCH, $RYCH$, $SABYCH$ and $IP5$ - ($\alpha = 1$, $\beta = 1$, $\gamma = \gamma_o = |M| * |F| * |N| * T$) - Sum of losses in machine qualifications

The percentage increase in the sum of completion times for YCH, $RYCH$ and $IP5$ with respect to the values obtained by $SABYCH$ is shown in Figure 5.17. $SABYCH$ dominates all approaches. $IP5$ is dominated by YCH and its derivatives. In addition, YCH and $RYCH$ alternatively dominate one the other, e.g. Instance types 19 and 22 where YCH dominates $RYCH$, and Instance type 38 where $RYCH$ dominates YCH. It is important to mention that YCH and its derivatives admit no feasible solution for In-

Table 6.b: Yield-Centric Heuristic (YCH) - ($\alpha' = 1$, $\beta' = 1$, $\gamma' = |M| * |F| * |N| * T$), $\sum_{f \in F} C_f$ and CPU times

	$\sum_{f\in F} C_f$				CPU Time			
No.	YCH	RYCH	SABYCH	IP5	YCH	RYCH	SABYCH	IP5
1	296	296	296	404	0	0	0	2.8
2	284	284	284	303	0	0	0	0.2
3	564	564	564	564	0	0	0	0.8
4	263	246	246	243	0	0	0	2.5
5	-	-	-	338	0	-	-	22.9
6	396	544	351	564	0	0	0	600
7	130	145	145	302	0	0	0	5.7
8	83	83	83	164	0	0	0	3.1
9	129	129	129	210	0	0	0	0.3
10	48	46	46	56	0	0	0	0.1
11	123	123	123	195	0	0	0	0.6
12	67	67	67	266	0	0	0	4.4
13	34	34	34	104	0	0	0	0.2
14	-	-	-	527	0	-	-	4.2
15	326	285	285	351	0	0	0	1.2
16	472	472	472	762	0	0	0	1.1
17	167	167	167	203	0	0	0	0.3
18	457	422	422	863	0	0	0	39.5
19	367	399	316	552	0	0	0	21.6
20	1216	1216	1216	1687	0	0	0	1.3
21	704	724	611	856	0	0	1	1.9
22	581	676	546	918	0	0	1	6.5
23	-	-	-	1141	0	-	-	69.3
24	716	674	646	1440	0	0	0	24.4
25	498	533	448	566	0	0	0	15.3
26	1148	1148	1148	4585	0	0	0	12.9
27	2435	2435	2427	3324	0	0	1	9.4
28	1668	1668	1286	2161	0	0	14	18.7
29	2069	2034	1790	2691	0	0	8	600
30	1746	1746	1655	2855	0	0	3	52
31	4133	4133	2636	2925	0	0	138	0.8
32	5429	5079	3376	3457	0	0	8	5.4
33	3259	3217	2762	4963	0	0	46	600
34	2927	2927	2927	6133	0	0	3	600
35	2126	2126	2082	4040	0	0	1	33.1
36	1850	1605	1561	3948	0	0	0	23.8
37	1337	1389	1325	4116	0	0	4	97
38	1181	1114	1042	2116	0	0	8	36.2
39	4432	4432	3623	6419	0	0	125	96.2
40	2973	2921	2794	5928	0	0	84	347.6
41	1282	1272	1201	4366	0	0	39	51.6
42	877	872	872	1944	0	0	56	37.6
43	3837	3837	3801	4768	0	0	216	33.9
44	3049	3142	2957	7047	0	0	514	600
45	2877	2877	2877	8248	0	0	0	91.5
46	3680	3671	3587	6340	0	0	40	600
47	3764	3766	3676	8460	0	0	7	425.5
48	2730	2834	2480	5046	0	0	369	293.2
49	2995	3015	2854	-	0	0	187	-
50	3114	3219	2897	-	0	0	203	-

stance type 14.

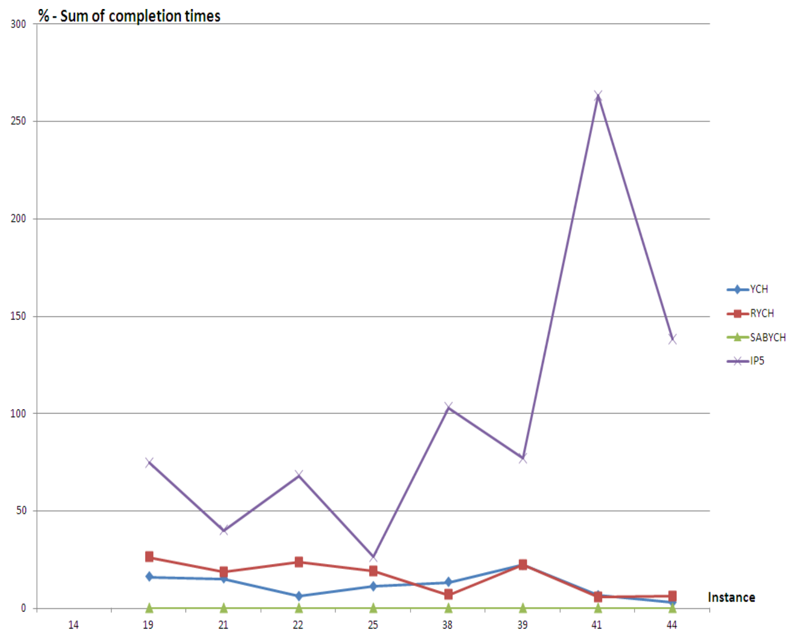


Figure 5.17: YCH, $RYCH$, $SABYCH$ and $IP5$ - ($\alpha = 1$, $\beta = 1$, $\gamma = \gamma_o = |M| * |F| * |N| * T$) - Sum of completion times

$IP5$ solves to optimality all the selected instances except one (Instance type 44) before the time limit. The chosen preference vector that does not prioritize machine qualifications led to CPU times lower than those obtained when minimizing machine qualifications due to the *relaxed* time constraints. $SABYCH$ attains large CPU times for large size instances (Instance types 39, 41 and 44). YCH and $RYCH$ solve all instances instantaneously.

5.7 Conclusion

In this chapter, numerical results obtained for solution approaches were shown. The results correspond to the problem of scheduling job families on non-identical parallel machines with time constraints PTC and the extension problem of scheduling with Equipment Health Factor $PEHF$. A multi-criteria objective function is considered. These criteria include: The sum of completion times, the loss in machine qualifications (PTC) and the expected yield ($PEHF$).

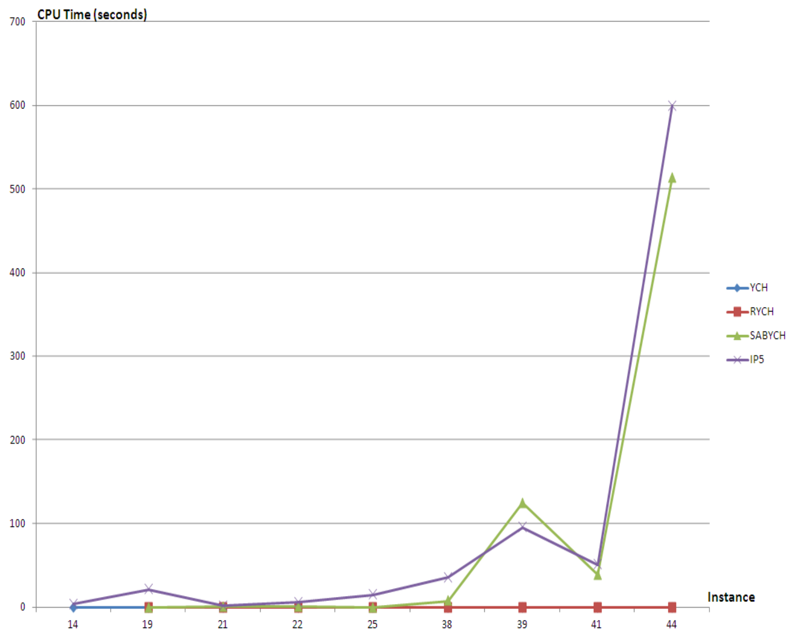


Figure 5.18: YCH, $RYCH$, $SABYCH$ and $IP5$ - ($\alpha = 1$, $\beta = 1$, $\gamma = \gamma_o = |M| * |F| * |N| * T$) - CPU times

We developed heuristics that target each criterion of the objective function, and numerical results on randomly generated instances were presented. These results showed, as expected, that the scheduling-centric heuristic gives generally better results regarding the sum of completion times, the qualification-centric heuristic provides better solutions on the number of machine qualification losses and the yield-centric heuristic has a better performance on the maximization of the expected yield. A Recursive Heuristic and a Simulated Annealing metaheuristic are able to provide effective solutions. The results of our heuristics were compared with exact solutions given by a standard solver. Simulated Annealing showed a remarkable performance when solving large size instances (number of jobs, families, machines, etc), and was able to reach optimum solutions in some cases for different criteria.

Algorithm 4 QCH pseudo code

```
1: $i = 1$
2: for $m = 1 \to |M|$ do
3:    $IDLE_m = TRUE$
4:    $t_m = 0$
5:    $f_m^* = 0$
6: end for
7: while $i \leq |N|$ do
8:    $Feasible = FALSE$
9:    $m = 1$
10:   for $m = 1 \to |M|$ do
11:      if $IDLE_m = TRUE$ then
12:         if $f_m^* \neq 0$ then
13:            - Select the family with the minimum threshold and that can be
               processed on $m$ ($f_m^*$)
14:            if $n_{f_m^*} \neq 0$ then
15:               $IDLE_m = FALSE$
16:               $C_{f_m^*} = UPDATE\_TIMES(f_m^*)$
17:               $t_m = t_m + p_{f_m^*}$
18:               $n_{f^*} \leftarrow n_{f_m^*} - 1$
19:               $Feasible = TRUE$
20:               $i = i + 1$
21:               - UPDATE_THRESHOLD($f_m^*, t_m, \gamma_F$)
22:               - Select $f_m'^*$ with $n_{f_m^*} > 0$ and with shortest current threshold
23:               if $f_m'^* \neq f_m^*$ then
24:                  $t_m = t_m + s_{f^*}$
25:               end if
26:               $f_m^* = f_m'^*$
27:            end if
28:         else
29:            $DISQUALIFY(m, f_m^*)$
30:            $y_{f_m^*}^m \leftarrow y_{f_m^*}^m + 1$
31:         end if
32:      end if
33:   end for
34:   for $m = 1 \to |M|$ do
35:      $IDLE_m = TRUE$
36:   end for
37:   if $bool = FALSE$ then
38:      print $No\ feasible\ solution.$
39:   end if
40: end while
41: $INTRACHANGE(Solution)$
42: $INTERCHANGE(Solution)$
```

Algorithm 5 YCH pseudo code

```
1: $i = 1$
2: for $m = 1 \rightarrow |M|$ do
3:    $IDLE_m = TRUE$
4:    $t_m = 0$
5:    $f^*_m = 0$
6: end for
7: while $i \leq |N|$ do
8:    $Feasible = FALSE$
9:    $m = 1$
10:   - Select the family with the maximum yield and that can be processed on
      $m$ ($f^*_m$)
11:   for $m = 1 \rightarrow |M|$ do
12:      if $IDLE_m = TRUE$ then
13:         if $f^*_m \neq 0$ then
14:            $min\_next = MIN\_NEXT(\gamma_F, f^*_m)$
15:            if $t_m < min\_next$ then
16:               if $n_{f^*_m} \neq 0$ then
17:                  $IDLE_m = FALSE$
18:                  $C_{f^*_m} = UPDATE\_TIMES(f^*_m)$
19:                  $t_m = t_m + p_{f^*_m}$
20:                  $n_{f^*_m} = n_{f^*_m} - 1$
21:                  $Feasible = TRUE$
22:                  $i = i + 1$
23:               end if
24:            else
25:               - Select $f^*_m$ with $n_{f^*_m} > 0$ and with maximum yield
26:               $t_m = t_m + s_{f^*}$
27:            end if
28:         else
29:            $DISQUALIFY(m, f^*_m)$
30:            $y^m_{f^*_m} = y^m_{f^*_m} + 1$
31:         end if
32:      end if
33:   end for
34:   for $m = 1 \rightarrow |M|$ do
35:      $IDLE_m = TRUE$
36:   end for
37:   if $Feasible = FALSE$ then
38:      print $No\ feasible\ solution.$
39:   end if
40: end while
```

Algorithm 6 Recursive Algorithm

```
1: get $INSTANCE_o$
2: $s = SOLVE(INSTANCE_o, HEURISTIC)$
3: $s^* = s$
4: if $s.DISQUALIFICATIONS \geq 1$ then
5:     $STOP\_ITER = 2^{s.DISQUALIFICATIONS}$
6:     $Counter = 1$
7:     while $Counter \leq STOP\_ITER$ do
8:         $INSTANCE = DISQUALIFY(INSTANCE_o, BINARY(Counter))$
9:         $s' = SOLVE(INSTANCE, HEURISTIC)$
10:        if $s'.SCORE < s^*.SCORE$ then
11:            $s^* = s'$
12:            $Counter = Counter + 1$
13:        end if
14:    end while
15: end if
16: RETURN $s^*$
```

Algorithm 7 SA pseudo code

```
1: $s = s_o$
2: $Set\ T(t_o)$
3: $t = T(t_o)$
4: while $t \neq 0$ do
5:     - Set $M_t$, the number of iteration at temperature $t$
6:     $Counter = 0$
7:     while $Counter \leq M_t$ do
8:         $Cost_s = SCORE(f, s)$
9:         $Cost_{s'} = NEIGHBOR(s)$
10:        $Cost_{s'} = SCORE(f, s')$
11:        if $Cost_{s'} \leq Cost_s$ then
12:            $s = s'$
13:        else
14:            $p_{s,s'} = RANDOM()$
15:            if $p_{s,s'} \leq \min(1, exp(-(Cost_s - Cost_{s'}))/t)$ then
16:                $s = s'$
17:            end if
18:        end if
19:        $Counter = Counter + 1$
20:    end while
21:    $t = DECREMENT(t)$
22: end while
23: $RETURN\ s$
```

General Conclusion and Perspectives

Conclusions

In this thesis, we discussed various possibilities of integrating scheduling decisions with information and constraints from Advanced Process Control (APC) systems in Semiconductor Manufacturing. In this context, important questions were opened regarding the benefits of integrating scheduling and APC. We partly answered these questions by proposing some ideas for integration and by developing novel approaches for new original problems: **Problem of Scheduling with Time Constraints** (*PTC*) and **Problem of Scheduling with Equipment Health Factor** (*PEHF*). *PTC* and *PEHF* have multicriteria objective functions. These problems were inspired from industrial needs and challenges. *PTC* aims at scheduling job in families on non-identical parallel machines with setup times and time constraints. Non-identical machines mean that not all machines can (are qualified to) process all types of job families, i.e. are non-identical in terms of qualification schemes. Time constraints related to *thresholds* are inspired from the needs of APC, for which APC control loops must be regularly fed with information from metrology operations (inspection) within a time interval (the *threshold*). The objective is to schedule job families on machines while minimizing the sum of completion times and the losses in machine qualifications. The complexity of this problem lies also in the fact that it is not only a decision to assign a machine to process a job, but also to select the appropriate family in order to keep control loop parameters updated and hence the machine qualified. We showed that this problem is NP-hard. *PEHF* is an extension of *PTC*, where job families are scheduled on non-identical parallel machines with time constraints and with Equipment Health Factors (*EHF*). *EHF* is an indicator on the state of a machine (e.g. poor, good, very good, excellent). The objective is to schedule jobs of different job families on machines while minimizing the sum of completion times, the losses in machine qualifications,

and maximizing the expected yield. This yield is defined as a function of the EHF and the criticality of the considered job. Hence, the scheduling decision in this problem is not only related to machine qualifications but also to an expected yield related to the states of both the machine and the job family.
An overview on processes, scheduling and Advanced Process Control in semiconductor manufacturing was done, where a description of semiconductor manufacturing processes is given. Also, scheduling issues and problems are outlined and a summary of Advanced Process Control components and techniques is provided. The past contributions on the integration of scheduling and Advanced Process Control were also presented. We discussed different integration issues, possibilities and perspectives and we proposed four major integration problems. Two of these problems (PTC and $PEHF$) were studied and analyzed in this thesis. A literature review concerning both problems was done, their complexity was addressed, and time indexed mixed integer linear programming models were proposed. Further, numerical experiments on these mathematical programming models were conducted. The results showed the strong compromise between the different criteria of the objective functions of both problems. Objective functions were considered as a weighted sum of different criteria. These weights are defined in preference vectors. Different types of preference vectors were studied and analyzed. A lexicographical order of criteria was considered. Moreover, an example of the ϵ-constraint method was also provided. A sensitivity study on the Time Constraint (*threshold*) was also performed. Numerical experiments showed limitations on the number of jobs, machines and families that a standard solver can handle. Therefore, several constructive heuristics and a metaheuristic (Simulated Annealing, SA) that address different criteria of the objective function were proposed. Dedicated heuristics, a recursive algorithm and SA showed a good performance on the related objectives.

Perspectives

Several research possibilities that may be studied as a continuation and/or extension of this thesis are given below.

Dynamic time thresholds

In PTC and $PEHF$, thresholds related to families are considered fixed over the time horizon and thus are time independent, i.e. $\gamma_{f,t} = \gamma_f =, \forall f \in F$,

$t = 1, \ldots, T$. However, it seems that a given family threshold $\gamma_{f,t_o} = \gamma_f$ could be dilated when the number of jobs of the family f processed during γ_f is greater than 1. In other words, when more jobs of the same family are processed during a given time interval γ_f, the degree of maturity of control loop parameters increases. This confidence in the parameters of the control loop can lead to consider that it is no longer a constant threshold γ_f that must be respected, but rather a dilated threshold $\gamma_{f,t}$. Hence, we propose to dynamically update family thresholds by considering the following equation:

$$\gamma_{f,t'} = \lambda * (\gamma_{f,t} - (t' - t)), \forall f \in F; t' = 1, \ldots, T; t = 1, \ldots, T; t' > t. \quad (5.1)$$

where $\lambda \geq 1$ is a real number, called the dilation factor, and is defined as a function of the number of processed jobs of a given family during its corresponding threshold. λ is strictly larger than 1 if and only if $n_{f,t'} < n_{f,t}$ and $t' < \gamma_{f,t} + t$, and is equal to 1 otherwise.

Different possible considerations may be distinguished for choosing λ. We mention four possible definitions:

- **Fixed λ.** The value of λ is independent of the families, and the machines. In this case, λ is predefined by the decision maker based on the number of processed jobs during a given threshold ($\lambda = k * n_f$, $\forall f \in F$, k is a natural number).

- **Family based λ (λ_f).** The value of λ is based on the family type. This means that there is no longer one value for all families, but rather a vector of values that contains the dilation factors of each family ($\lambda_f = k_f * n_f$, k_f is a natural number). The values of λ_f may be different for each family to relate to its criticality (related to Lot Criticality Indicator (LCI)) addressed in Chapter 2.

- **Machine based λ (λ_m).** The value of λ is based on the machine where the job/lot is scheduled ($\lambda_m = k_m * n_f$, k_m is a natural number). By considering such a vector of λ, we involve the notion of dynamic machine reliability where a more reliable machine may have dilation factors greater than other less reliable machines.

- **Mixed λ ($\lambda_{f,m}$).** The value of λ is based on both the family and the machine ($\lambda_{f,m} = k_{f,m} * n_f$, $k_{f,m}$ is a natural number). This involves both aspects related to family criticality and machine reliability.

Dynamic yield

The notion of dynamic time thresholds can be applied to $PEHF$, where the values of the expected yield corresponding to assigning a machine m to process a job of family f may change as a function of time. Let us recall that these values are defined based on the machine state and the family criticality. Hence, $v_f^m = f(\text{EHF,LCI})$ may become $v_{f,t}^m = f(\text{EHF,LCI}, t)$. However, the value of the dynamic expected yield depends on the results of a metrology operation, i.e. when the results of sampled and tested lots in metrology that were processed on a given machine m are "to specifications" at a period t, then the value of the expected yield increases at period t. This is related to the notion of Wafer at Risk (W@R) (Dauzère-Pérès et al. (2010)), where the W@R is the number of wafers between two control operations.

Priorities on job families

Priorities on job families regarding delivery times can be included by considering, instead of the sum of completion times, a weighted sum of completion times $\sum_{f\in F} w_f C_f$, where w_f presents the weight/penalty associated to family f. For example, if family f needs to be finished before family f', then w_f must be greater than $w_{f'}$. This also can be extended to the sum of losses in machine qualifications (PTC) and the sum of expected yield ($PEHF$), hence the objective function of PTC, to be minimized, becomes:

$$\sum_{f\in F} w_f(\alpha C_f + \beta \sum_{m\in M} Y_f^m) \tag{5.2}$$

And that of $PEHF$, to be maximized, becomes:

$$\sum_{f\in F} w_f(\sum_{m\in M} (\gamma' \sum_{t=1}^{T} x_{f,t}^m v_{f,t}^m - \beta' Y_f^m) - \alpha' C_f) \tag{5.3}$$

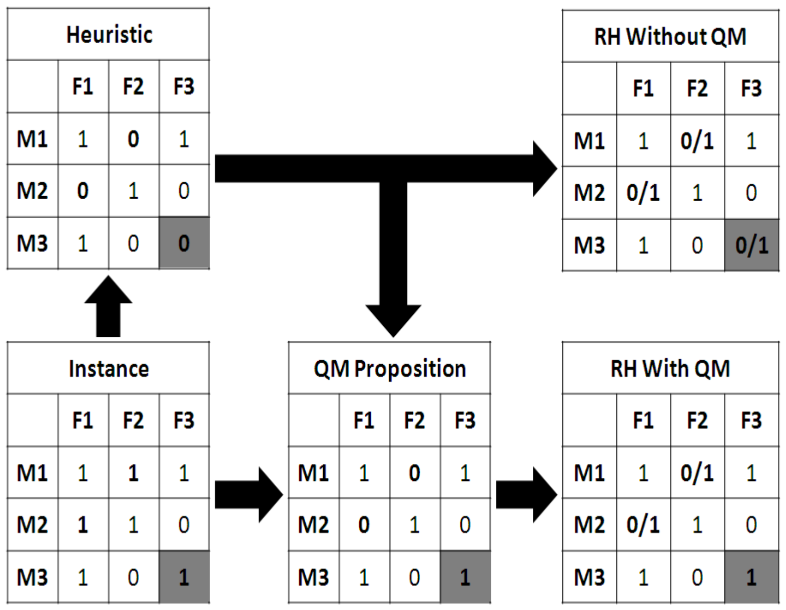
Integrating flexibility, scheduling and qualifications

Qualification Management (QM) is studied in a previous thesis in our laboratory (Johnzén (2009)). In this work, the interaction between Qualification Management policies and flexibility measures such as Work In Process (WIP) flexibility, time flexibility, toolset flexibility, system flexibility, are studied (Johnzén et al. (2010)). The impact of QM on scheduling is also addressed in this work. In our perspectives, we believe that it is possible to integrate the results of Johnzén (2009) in PTC and $PEHF$. In fact, the proposed qualification/disqualification schemes can be integrated by enabling machine re-qualifications in our problems, i.e. when a machine loses its qualification, we then check whether the disqualified machine should stay qualified according to the proposed qualification scheme of the QM system (to keep on appropriate flexibility measures). If this is the case, we then set a qualification run. Send Ahead Wafers (SAW) are usually used to test machine qualification in semiconductor manufacturing. These wafers are non-productive wafers and are usually scrapped. The cost of re-qualifying the machine in this case may be illustrated by the gain in flexibility and even in the throughput of the machine. This is a crucial in high mix/low volume semiconductor manufacturing facilities. Machine re-qualification usually includes different types of qualifications, i.e. we may find "easy" re-qualifications (do not require many resources and/or time), "normal" re-qualifications (that are done using some resources and acceptable configuration times), and last but not least "hard" re-qualifications (that require special resources and long times). These last re-qualifications are often skipped or postponed. These types help also in proposing machine re-qualifications for families and machine disqualifications if flexibility measures are prioritized. In this context, a new parameter can be defined, namely the re-qualification threshold (∇_f^m), which indicates the time period by which a machine m should be re-qualified to process job family f. Hence, relating to $IP3$ and $IP5$, we add the following constraint:

$$x_{f,t}^m \geq y_{f,t}^m, \forall f \in F, \forall m \in M, t = \nabla_f^m, \ldots, T \tag{5.4}$$

Application on Recursive Heuristic (RH)

An Enhanced Recursive Heuristic (ERH) can be proposed. Let us recall that the recursive algorithm proposed in this thesis considers the perturbations on qualification schemes. An infeasible qualification of a constructive heuristic solution enables a perturbation in the initial qualification scheme of an instance, followed by re-applying the considered constructive heuristic. The complexity here lies in the case of two or more infeasible qualifications. The number of combinations of perturbations is 2^{IQ}, where IQ is the number of infeasible qualifications. Hence, the question is which combination to choose and what machines for which families should be disqualified or kept qualified. Here, the proposed qualification schemes by Johnzén (2009) are very helpful to answer these questions. An example on the effect of integrating QM in RH is shown in Figure 5.19. Note that the number of choices on family and machine to disqualify is reduced from 8 possibilities in RH without QM to 4 possibilities in RH with QM.



Heuristic

	F1	F2	F3
M1	1	0	1
M2	0	1	0
M3	1	0	0

RH Without QM

	F1	F2	F3
M1	1	0/1	1
M2	0/1	1	0
M3	1	0	0/1

Instance

	F1	F2	F3
M1	1	1	1
M2	1	1	0
M3	1	0	1

QM Proposition

	F1	F2	F3
M1	1	0	1
M2	0	1	0
M3	1	0	1

RH With QM

	F1	F2	F3
M1	1	0/1	1
M2	0/1	1	0
M3	1	0	1

Figure 5.19: Example on integrating QM with RH

Other Perspectives

Other perspectives can be proposed on different aspects of this thesis. For example, further complexity analysis for both PTC and $PEHF$ can be done by considering the strong sense or weak sense of the NP-hard studied problems. Exact methods for larger instances may be developed as for example a column generation approach. Other solution approaches may be tested as for example Genetic Algorithms or hybrid math-heuristics. In addition, further multi-criteria optimization methods can be used to thoroughly analyze the antagonistic nature of the conflicting criteria. Other dimensions can be added to scheduling and Advanced Process Control as for example considering design constraints and needs through APC and scheduling and vice-versa. For instance, some new components to integrate in the design may require fully qualified machines and hence a qualification run should be forced by sending a Send Ahead Wafer to verify machine parameters which affect both scheduling and APC.

Appendix A

Simulation on real data

Effect of threshold variation with real data

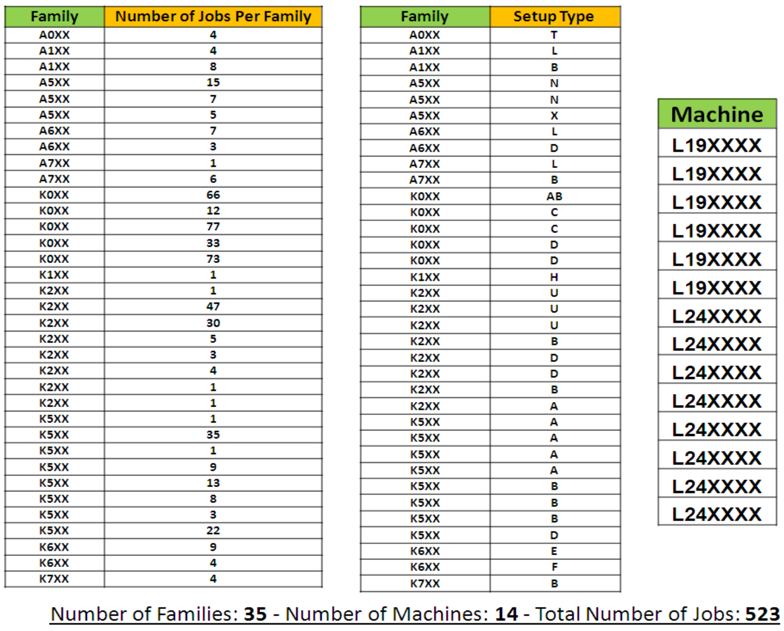
In this section, we address a test on real data. The pre-processing of data is presented. The effect of threshold variation on both criteria of *PTC* is studied.

Pre-processing of data

Industrial data are collected from a real industry. These data needed several steps of pre-processing to be adapted to our problem. After pre-treating the data, we end up with an instance of 523 jobs, 35 machines and 14 families. It is obvious that this instance is too large to be treated by a standard solver and therefore we apply our heuristics for *PTC* and *SA*, to study the effect of threshold variation on the number of losses in machine qualifications and the sum of completion times and to compare the performance of our approaches when solving an industrial instance. Figure A.1 gives an example of data preparation.

Processing time calculation

In our problem, the processing time per family is constant for all jobs. However, it is not exactly the case in the real data where processing times can slightly differ from one job to another in the same family. This is why we use the average value of processing times of all jobs which belong to the same family, and hence the processing time is unified. Figure A.2 gives an example on how processing times are estimated from a real sample. The names of jobs/lots are omitted for confidentiality reasons.



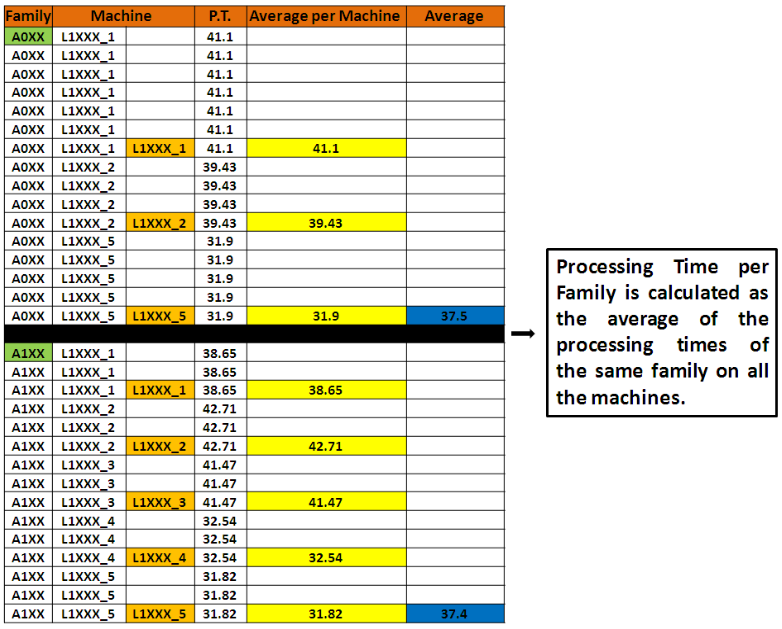
Family	Number of Jobs Per Family
A0XX	4
A1XX	4
A1XX	8
A5XX	15
A5XX	7
A5XX	5
A6XX	7
A6XX	3
A7XX	1
A7XX	6
K0XX	66
K0XX	12
K0XX	77
K0XX	33
K0XX	73
K1XX	1
K2XX	1
K2XX	47
K2XX	30
K2XX	5
K2XX	3
K2XX	4
K2XX	1
K2XX	1
K5XX	1
K5XX	35
K5XX	1
K5XX	9
K5XX	13
K5XX	8
K5XX	3
K5XX	22
K6XX	9
K6XX	4
K7XX	4

Family	Setup Type
A0XX	T
A1XX	L
A1XX	B
A5XX	N
A5XX	N
A5XX	X
A6XX	L
A6XX	D
A7XX	L
A7XX	B
K0XX	AB
K0XX	C
K0XX	C
K0XX	D
K0XX	D
K1XX	H
K2XX	U
K2XX	U
K2XX	U
K2XX	B
K2XX	D
K2XX	D
K2XX	B
K2XX	A
K5XX	A
K5XX	A
K5XX	A
K5XX	A
K5XX	B
K5XX	B
K5XX	B
K5XX	D
K6XX	E
K6XX	F
K7XX	B

Machine
L19XXXX
L19XXXX
L19XXXX
L19XXXX
L19XXXX
L19XXXX
L24XXXX
L24XXXX
L24XXXX
L24XXXX
L24XXXX
L24XXXX
L24XXXX
L24XXXX

Number of Families: **35** - Number of Machines: **14** - Total Number of Jobs: **523**

Figure A.1: Preparing data



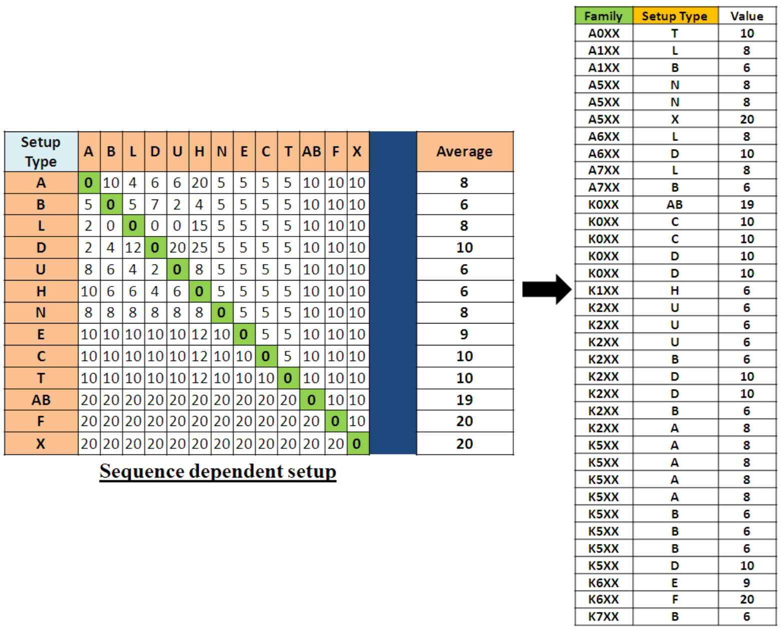
Family	Machine		P.T.	Average per Machine	Average
A0XX	L1XXX_1		41.1		
A0XX	L1XXX_1		41.1		
A0XX	L1XXX_1		41.1		
A0XX	L1XXX_1		41.1		
A0XX	L1XXX_1		41.1		
A0XX	L1XXX_1		41.1		
A0XX	L1XXX_1	L1XXX_1	41.1	41.1	
A0XX	L1XXX_2		39.43		
A0XX	L1XXX_2		39.43		
A0XX	L1XXX_2		39.43		
A0XX	L1XXX_2	L1XXX_2	39.43	39.43	
A0XX	L1XXX_5		31.9		
A0XX	L1XXX_5		31.9		
A0XX	L1XXX_5		31.9		
A0XX	L1XXX_5		31.9		
A0XX	L1XXX_5	L1XXX_5	31.9	31.9	37.5
A1XX	L1XXX_1		38.65		
A1XX	L1XXX_1		38.65		
A1XX	L1XXX_1	L1XXX_1	38.65	38.65	
A1XX	L1XXX_2		42.71		
A1XX	L1XXX_2		42.71		
A1XX	L1XXX_2	L1XXX_2	42.71	42.71	
A1XX	L1XXX_3		41.47		
A1XX	L1XXX_3		41.47		
A1XX	L1XXX_3	L1XXX_3	41.47	41.47	
A1XX	L1XXX_4		32.54		
A1XX	L1XXX_4		32.54		
A1XX	L1XXX_4	L1XXX_4	32.54	32.54	
A1XX	L1XXX_5		31.82		
A1XX	L1XXX_5		31.82		
A1XX	L1XXX_5	L1XXX_5	31.82	31.82	37.4

$\rightarrow$ Processing Time per Family is calculated as the average of the processing times of the same family on all the machines.

Figure A.2: Calculating processing times

Setup times calculation

Setup times were taken as sequence independent. This means that a shift from one job of family f to one job of family f' requires the same setup time as in the case of shifting from f' to f. It is not the case in industrial data (see Figure A.3). Hence, we make a hypothesis which is to take the average value of setup times from a certain family f to all other families. The obtained value is considered as the setup time of the considered family.



Setup Type	A	B	L	D	U	H	N	E	C	T	AB	F	X		Average
A	0	10	4	6	6	20	5	5	5	5	10	10	10		8
B	5	0	5	7	2	4	5	5	5	5	10	10	10		6
L	2	0	0	0	0	15	5	5	5	5	10	10	10		8
D	2	4	12	0	20	25	5	5	5	5	10	10	10		10
U	8	6	4	2	0	8	5	5	5	5	10	10	10		6
H	10	6	6	4	6	0	5	5	5	5	10	10	10		6
N	8	8	8	8	8	8	0	5	5	5	10	10	10		8
E	10	10	10	10	10	12	10	0	5	5	10	10	10		9
C	10	10	10	10	10	12	10	10	0	5	10	10	10		10
T	10	10	10	10	10	12	10	10	10	0	10	10	10		10
AB	20	20	20	20	20	20	20	20	20	20	0	10	10		19
F	20	20	20	20	20	20	20	20	20	20	20	0	10		20
X	20	20	20	20	20	20	20	20	20	20	20	20	0		20

Sequence dependent setup

Family	Setup Type	Value
A0XX	T	10
A1XX	L	8
A1XX	B	6
A5XX	N	8
A5XX	N	8
A5XX	X	20
A6XX	L	8
A6XX	D	10
A7XX	L	8
A7XX	B	6
K0XX	AB	19
K0XX	C	10
K0XX	C	10
K0XX	D	10
K0XX	D	10
K1XX	H	6
K2XX	U	6
K2XX	U	6
K2XX	U	6
K2XX	B	6
K2XX	D	10
K2XX	D	10
K2XX	B	6
K2XX	A	8
K5XX	A	8
K5XX	A	8
K5XX	A	8
K5XX	A	8
K5XX	B	6
K5XX	B	6
K5XX	B	6
K5XX	D	10
K6XX	E	9
K6XX	F	20
K7XX	B	6

Figure A.3: Calculating setup times - From sequence dependent to sequence independent

Qualification matrix

The total number of qualified machines for all families is 129. This means that a solution based on the number of losses in machine qualifications can be compared with 129 in order to test algorithms in solving the industrial instance in terms of qualifications.

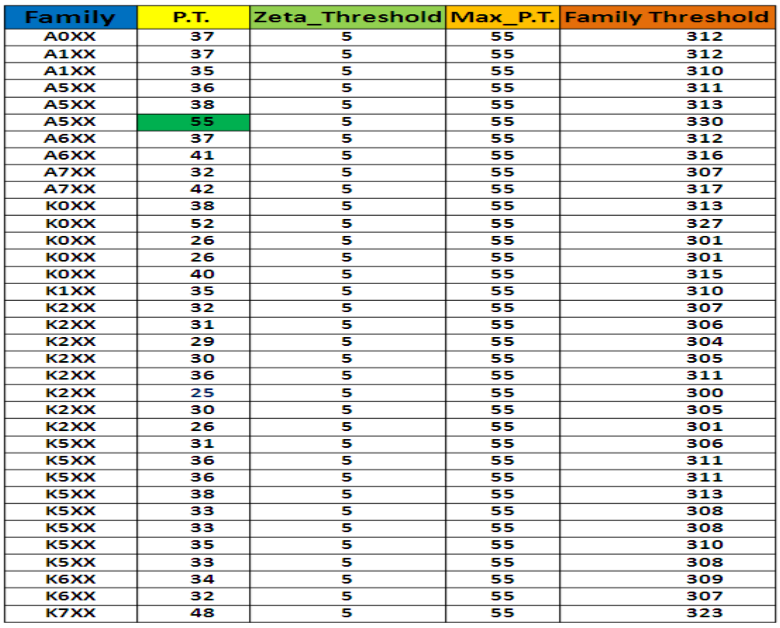
Results on threshold variation

Thresholds were not given by the industry. We generated thresholds based on the following equation:

$$\gamma_f = (\zeta * \max_{f \in F} p_f) + p_f \tag{A.1}$$

where, ζ is a coefficient to vary the threshold value, $\zeta \in [1,2\ldots,10]$, and $\max_{f \in F} p_f$ is the largest processing time for all families.

Tests are done on this instance by applying LH, SCH and SA, and the results are shown in Table A.1. Note that for larger values of ζ, the time threshold per family increases and the number of losses in machine qualifications decreases. This is expected since, when increasing family thresholds, it is easier to keep machines qualified. SA dominates in all cases and for both criteria of the objective function.



Family	P.T.	Zeta_Threshold	Max_P.T.	Family Threshold
A0XX	37	5	55	312
A1XX	37	5	55	312
A1XX	35	5	55	310
A5XX	36	5	55	311
A5XX	38	5	55	313
A5XX	55	5	55	330
A6XX	37	5	55	312
A6XX	41	5	55	316
A7XX	32	5	55	307
A7XX	42	5	55	317
K0XX	38	5	55	313
K0XX	52	5	55	327
K0XX	26	5	55	301
K0XX	26	5	55	301
K0XX	40	5	55	315
K1XX	35	5	55	310
K2XX	32	5	55	307
K2XX	31	5	55	306
K2XX	29	5	55	304
K2XX	30	5	55	305
K2XX	36	5	55	311
K2XX	25	5	55	300
K2XX	30	5	55	305
K2XX	26	5	55	301
K5XX	31	5	55	306
K5XX	36	5	55	311
K5XX	36	5	55	311
K5XX	38	5	55	313
K5XX	33	5	55	308
K5XX	33	5	55	308
K5XX	35	5	55	310
K5XX	33	5	55	308
K6XX	34	5	55	309
K6XX	32	5	55	307
K7XX	48	5	55	323

Figure A.4: Defining family thresholds

Table A.1: Results of heuristics and simulated annealing applied on real data (best solution in bold).

Results obtained upon using real data						
	LH		SCH		SA - (1,$\|N\| * T$)	
ζ	$\sum_{f \in F} C_f(days)$	$\sum_{f \in F} \sum_{m \in M} Y_f^m$	$\sum_{f \in F} C_f(days)$	$\sum_{f \in F} \sum_{m \in M} Y_f^m$	$\sum_{f \in F} C_f(days)$	$\sum_{f \in F} \sum_{m \in M} Y_f^m$
1	**4**	126	83	119	83	**114**
2	399	118	**19**	112	**19**	**110**
3	432	111	188	107	**185**	**105**
4	329	107	200	200	**199**	**98**
5	**264**	99	469	**94**	468	**94**
6	260	96	504	97	**503**	**96**
7	**242**	92	514	91	505	**89**
8	563	87	376	87	**374**	**85**
9	475	83	271	**83**	**270**	**83**
10	481	80	**353**	**79**	**353**	**79**

Bibliography

S. Albers. The complexity of one-machine batching problems. *Discrete Applied Mathematics*, 47(2):87–107, 1993. ISSN 0166218X.

A. Allahverdi, C. T. Ng, T. C. E. Cheng, and M. Y. Kovalyov. A survey of scheduling problems with setup times or costs. *European Journal of Operational Research*, 187(3):985–1032, 2008. ISSN 0377-2217.

G. C. Anagnostopoulos and G. Rabadi. A simulated annealing algorithm for the unrelated parallel machine scheduling problem. In *Proceedings of the 5th Biannual World Automation Congress 2002*, volume 14, pages 115–120, 2002.

J. H. Anderson. *Handbook of Scheduling: Algorithms, Models, and Performance Analysis.* Chapman and Hall/CRC, 1 edition, 2004. ISBN 1584883979.

M. Anderson and C. K. Hanish. An evaluation of the benefits of integrating run-to-run control with scheduling and dispatching systems. *IEEE Transactions on Semiconductor Manufacturing*, 20(4):386–392, 2007. ISSN 0894-6507.

A. Bousetta and A. J. Cross. Adaptive sampling methodology for in-line defect inspection. In *IEEE/SEMI Advanced Semiconductor Manufacturing Conference and Workshop 2005*, pages 25–31, 2005.

P. Brucker, S. Knust, A. Schoo, and O. Thiele. A branch and bound algorithm for the resource-constrained project scheduling problem. *European Journal of Operational Research*, 107(2):272–288, 1998. ISSN 0377-2217.

M. Caccamo, G. Lipari, and G. Buttazzo. Sharing resources among periodic and aperiodic tasks with dynamic deadlines. In *Proceedings of IEEE International Real-Time Systems Symposium 1999*, pages 284–293, 1999. doi: 10.1109/REAL.1999.818856.

Y. Cai, E. Kutanoglu, J. Hasenbein, and J. Qin. Single-machine scheduling with advanced process control constraints. *Journal of Scheduling*, pages 1–15, 2011. ISSN 1094-6136.

J. Carlier and P. Chrétienne. *Problèmes d'Ordonnancement: Modélisation, Complexité, Algorithmes.* Masson, 1988. ISBN 9782225812750.

C. R. Cassady and E. Kutanoglu. Integrating preventive maintenance planning and production scheduling for a single machine. *IEEE Transactions on Reliability*, 54(2):304–309, 2005. ISSN 0018-9529.

G. Centeno and R. L. Armacost. Parallel machine scheduling with release time and machine eligibility restrictions. *Computers and Industrial Engineering*, 33(1-2):273–276, 1997. ISSN 0360-8352.

Y. J. Chang, Y. Kang, C. L. Hsu, C. T. Chang, and T. Y. Chan. Virtual metrology technique for semiconductor manufacturing. In *International Joint Conference on Neural Networks 2006*, pages 5289–5293, 2006.

A. Chen and G. S. Wu. Real-time health prognosis and dynamic preventive maintenance policy for equipment under aging markovian deterioration. *International Journal of Production Research*, 45(15):3351–3379, 2007. ISSN 0020-7543.

P. H. Chen, S. Wu, J. Lin, F. Ko, H. Lo, J. Wang, C. H. Yu, and M. S. Liang. Virtual metrology: A solution for wafer to wafer advanced process control. In *IEEE International Symposium on Semiconductor Manufacturing (ISSM) 2005*, pages 155–157, 2005.

F. T. Cheng, H. C. Huang, and C. A. Kao. Dual-Phase virtual metrology scheme. *IEEE Transactions on Semiconductor Manufacturing*, 20(4):566–571, 2007. ISSN 0894-6507.

C. A. C. Coello. Evolutionary multiobjective optimization. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1(5):444–447, 2011. ISSN 1942-4795.

S. Dauzère-Pérès. Recherche opérationnelle et aide à la décision en fabrication de semi-conducteurs. *ROADEF - Le Bulletin*, 27:8–11, 2011.

S. Dauzère-Pérès, J.L. Rouveyrol, C. Yugma, and P. Vialletelle. A smart sampling algorithm to minimize risk dynamically. In *IEEE/SEMI Advanced Semiconductor Manufacturing Conference 2010*, pages 307–310, 2010.

K. Deb. *Multi-Objective Optimization Using Evolutionary Algorithms.* John Wiley & Sons Ltd, 2001. ISBN 047187339X.

B. Detienne, S. Dauzère-Pérès, and C. Yugma. Scheduling jobs on parallel machines to minimize a regular step total cost function. *Journal of Scheduling*, 14(6).

B. Detienne, S. Dauzère-Pérès, and C. Yugma. An exact approach for scheduling jobs with regular step cost functions on a single machine. *Computers and Operations Research*, 39(5):1033–1043, 2012. ISSN 0305-0548.

T. F. Edgar, S. W. Butler, W. J. Campbell, C. Pfeiffer, C. Bode, S. B. Hwang, K. S. Balakrishnan, and J. Hahn. Automatic control in microelectronics manufacturing: Practices, challenges, and possibilities. *Automatica*, 36 (11):1567–1603, 2000. ISSN 0005-1098.

R. W. Eglese. Simulated annealing: A tool for operational research. *European Journal of Operational Research*, 46(3):271–281, 1990. ISSN 0377-2217.

M. Ehrgott and X. Gandibleux. A survey and annotated bibiliography of multiobjective combinatorial optimization. *OR Spektrum*, 22:425–460, 2000.

D.-H. Eom, H.-J. Shin, I.-H. Kwun, J.-K. Shim, and S.-S. Kim. Scheduling jobs on parallel machines with sequence-dependent family set-up times. *The International Journal of Advanced Manufacturing Technology*, 19:926–932, 2002. ISSN 0268-3768.

H. Eschenauer, J. Koski, and A. Osyczka. *Multicriteria Design Optimization: Procedures and Applications.* Springer-Verlag, 1990. ISBN 9780387506043.

A. Faruqi, R. Goss, D. Adhikari, and T. Kowtsch. Test wafer management and automated wafer sorting. In *IEEE/SEMI Advanced Semiconductor Manufacturing Conference 2008*, pages 322–326, 2008.

J. W. Fowler, N. Phojanamongkolkij, J. K. Cochran, and D. C. Montgomery. Optimal batching in a wafer fabrication facility using a multiproduct model with batch processing. *International Journal of Production Research*, 40 (2):275–292, 2002. ISSN 0020-7543.

R. L. Graham, E. L. Lawler, J. K. Lenstra, and A. H. G. Rinnooy Kan. Optimization and approximation in deterministic sequencing and scheduling: a survey. volume 5, pages 287–326. Elsevier, 1979.

R. S. Guo, A. Chen, C. L. Tseng, I. K. Fong, A. Yang, C. L. Lee, C. H. Wu, S. Lin, S. J. Huang, Y. C. Lee, S. G. Chang, and M. Y. Lee. A real-time equipment monitoring and fault detection system. In *Proceedings of Semiconductor Manufacturing Technology Workshop 1998*, pages 111–121, 1998.

Y. Guo and X. Liu. Scheduling uniform parallel machines with machine eligibility restrictions to minimize total weighted tardiness. In *Proceedings of International Conference on Computer Application and System Modeling 2010*, volume 8, pages 165–168, 2010.

A. Gupta and A. I. Sivakumar. Job shop scheduling techniques in semiconductor manufacturing. *The International Journal of Advanced Manufacturing Technology*, 27(11-12):1163–1169, 2004. ISSN 0268-3768.

A. K. Gupta and A. I. Sivakumar. Job shop scheduling techniques in semiconductor manufacturing. *The International Journal of Advanced Manufacturing Technology*, 27:1163–1169, 2006. ISSN 0268-3768.

Q. P. He and J. Wang. Fault detection using the k-Nearest neighbor rule for semiconductor manufacturing processes. *IEEE Transactions on Semiconductor Manufacturing*, 20(4):345–354, 2007. ISSN 0894-6507.

A. Holfeld, R. Barlovic, and R. P. Good. A fab-wide apc sampling application. *IEEE Transactions on Semiconductor Manufacturing*, 20(4):393 –399, 2007. ISSN 0894-6507.

R. H. Huang and T. H. Yu. An effective heuristic considering machine flexibility for parallel machine with eligibility problem. In *Proceedings of IEEE International Conference on Industrial Engineering and Engineering Management 2010*, pages 1363–1366, 2010.

S. Hubac, F. Duvivier, E. Zamai, and A. Mili. Predictive maintenance supported by advanced process control (apc) opens new equipment engineering and manufacturing opportunities. *SEMI/IEEE Advanced Semiconductor Manufacturing Conference 2010*, 2010.

M. H. Hung, T. H. Lin, F. T. Cheng, and R. C. Lin. A novel virtual metrology scheme for predicting CVD thickness in semiconductor manufacturing. *IEEE/ASME Transactions on Mechatronics*, 12(3):308–316, 2007. ISSN 1083-4435.

N. Jedidi. Applications de techniques avancées de contrôle des procédés en industrie du semi-conducteur, 2009.

C. Johnzén. Allocation flexible des capacités pour la fabrication de semi-conducteurs : Modélisation et optimisation, 2009.

C. Johnzén, S. Dauzère-Pérès, and P. Vialletelle. Flexibility measures for qualification management in wafer fabs. *Production Planning & Control*, 22(1):81–90, 2010. ISSN 0953-7287.

P. V. Kallapur and N. N. Chiplunkar. Topology aware mobile agent for efficient data collection in wireless sensor networks with dynamic deadlines. In *Proceedings of International Conference on Advances in Computer Engineering (ACE) 2010*, pages 352–356, 2010.

A. A. Khan, J. R. Moyne, and D. M. Tilbury. An approach for Factory-Wide control utilizing virtual metrology. *IEEE Transactions on Semiconductor Manufacturing*, 20(4):364–375, 2007. ISSN 0894-6507.

B. Kim and G. S. May. Real-time diagnosis of semiconductor manufacturing equipment using a hybrid neural network expert system. *IEEE Transactions on Components, Packaging, and Manufacturing Technology, Part C*, 20(1):39–47, 1997. ISSN 1083-4400.

D. W. Kim, K. H. Kim, W. Jang, and F. F. Chen. Unrelated parallel machine scheduling with setup times using simulated annealing. *Robotics and Computer-Integrated Manufacturing*, 18(3-4):223–231, 2002. ISSN 0736-5845.

Y. D. Kim, J. U. Kim, S. K. Lim, and H. B. Jun. Due-date based scheduling and control policies in a multiproduct semiconductor wafer fabrication facility. *IEEE Transactions on Semiconductor Manufacturing*, 11(1):155–164, 1998. ISSN 0894-6507.

S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

W. Kubiak, S. X. C. Lou, and Y. Wang. Mean flow time minimization in reentrant job shops with a hub. *Operations Research*, 44(5):764–776, 1996. ISSN 0030364X.

E. D. W. Kum. *One Machine Generalized Precedence Constrained Scheduling.* PhD thesis, School of Industrial Systems Engineering, Georgia Institute of Technology, 1992.

P. R. Kumar. Re-Entrant lines. *Queueing Systems*, 13:87–110, 1993.

S. B. Lee, T. Y. Lee, J. Liao, and Y. C. Chang. A capacity-dependence dynamic sampling strategy. In *IEEE International Symposium on Semiconductor Manufacturing 2003*, pages 312–314, 2003.

T. E. Lee. A review of scheduling theory and methods for semiconductor manufacturing cluster tools. In *Winter Simulation Conference 2008*, pages 2127–2135, 2008.

L. Li and F. Qiao. The impact of the qual-run requirements of APC on the scheduling performance in semiconductor manufacturing. In *IEEE International Conference on Automation Science and Engineering 2008*, pages 242–246, 2008.

L.-W. Liao and G.-J. Sheen. Parallel machine scheduling with machine availability and eligibility constraints. *European Journal of Operational Research*, 184(2):458–467, 2008. ISSN 0377-2217.

T. H. Lin, M. H. Hung, R. C. Lin, and F. T. Cheng. A virtual metrology scheme for predicting CVD thickness in semiconductor manufacturing. In *Proceedings of IEEE International Conference on Robotics and Automation 2006*, pages 1054–1059, 2006. ISBN 1050-4729.

Y. K. Lin, M. E. Pfund, and J. W. Fowler. Minimizing makespans for unrelated parallel machine scheduling problems. In *IEEE/INFORMS International Conference on Service Operations, Logistics and Informatics 2009*, pages 107–110, 2009.

S. J. Mason, J. W. Fowler, and W. M. Carlyle. A modified shifting bottleneck heuristic for minimizing total weighted tardiness in complex job shops. *Journal of Scheduling*, 5(3):247–262, 2002. ISSN 1094-6136.

M. Mathirajan and A. I. Sivakumar. A literature review, classification and simple meta-analysis on scheduling of batch processors in semiconductor. *The International Journal of Advanced Manufacturing Technology*, 29:990–1001, 2006. ISSN 0268-3768.

G. S. May and S. M. Sze. *Fundamentals of Semiconductor Fabrication*. Wiley, 1st edition, 2003. ISBN 0471232793.

N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953. ISSN 00219606.

Kaisa Miettinen. *Nonlinear Multiobjective Optimization*. Springer, 1999. ISBN 9780792382782.

L. Moench, J. W. Fowler, S. Dauzère-Pérès, S. J. Mason, and O. Rose. A survey of problems, solution techniques, and future challenges in scheduling semiconductor manufacturing operations. *Journal of Scheduling*, 14:583–599, 2011. ISSN 1094-6136.

D. C. Montgomery. *Introduction to Statistical Quality Control*. Wiley, 5 edition, 2004. ISBN 0471656313.

J. Moyne, E. del Castillo, and A. M. Hurwitz. *Run-to-Run Control in Semiconductor Manufacturing*. CRC Press, 1 edition, 2000. ISBN 0849311780.

J. Musacchio, S. Rangan, C. Spanos, and K. Poolla. On the utility of run to run control in semiconductor manufacturing. In *Proceedings of IEEE International Symposium on Semiconductor Manufacturing Conference 1997*, pages D9–12, 1997.

D. G. Na, D. W. Kim, W. Jang, and F. F. Chen. Scheduling unrelated parallel machines to minimize total weighted tardiness. In *IEEE International Conference on Service Operations and Logistics, and Informatics 2006*, pages 758–763, 2006.

M. L. Pinedo. *Planning and Scheduling in Manufacturing and Services*. Springer, 2009. ISBN 9781441909091.

C. N. Potts and M. Y. Kovalyov. Scheduling with batching: A review. *European Journal of Operational Research*, 120(2):228–249, 2000. ISSN 0377-2217.

C. N. Potts and L. N. Van Wassenhove. Integrating scheduling with batching and lot-sizing: A review of algorithms and complexity. *The Journal of the Operational Research Society*, 43(5):395–406, 1992. ISSN 01605682.

M. Purdy. Dynamic, weight-based sampling algorithm. In *International Symposium on Semiconductor Manufacturing 2007*, pages 1–4, 2007.

S. J. Qin, G. Cherry, R. Good, J. Wang, and C. A. Harrison. Semiconductor manufacturing process control and monitoring: A fab-wide framework. *Journal of Process Control*, 16(3):179–191, 2006. ISSN 0959-1524.

M. Razzaghi and R. Kodell. Quantitative risk assessment for developmental neurotoxic effects. *Risk Anal*, 24(6):1673–81, 2004. ISSN 0272-4332.

R. Ruiz and C. Andrés. Scheduling unrelated parallel machines with resource-assignable sequence dependent setup times, 2007.

J. E. Schaller, J. N. D. Gupta, and A. J. Vakharia. Scheduling a flowline manufacturing cell with sequence dependent family setup times. *European Journal of Operational Research*, 125(2):324–339, 2000. ISSN 0377-2217.

G.-J. Sheen, L.-W. Liao, and C.-F. Lin. Optimal parallel machines scheduling with machine availability and eligibility constraints. *The International Journal of Advanced Manufacturing Technology*, 36:132–139, 2008. ISSN 0268-3768.

Donald E. Shobrys and Douglas C. White. Planning, scheduling and control systems: Why cannot they work together. *Computers and Chemical Engineering*, 26:149–160, 2002.

A. Somasundara, A. Ramamoorthy, and M. Srivastava. Mobile element scheduling with dynamic deadlines. *IEEE Transactions on Mobile Computing*, 6(4):395–410, 2007. ISSN 1536-1233.

Y. C. Su, T. H. Lin, F. T. Cheng, and W. M. Wu. Accuracy and Real-Time considerations for implementing various virtual metrology algorithms. *IEEE Transactions on Semiconductor Manufacturing*, 21(3):426–434, 2008. ISSN 0894-6507.

J. Teghem. *Optimisation Approchée en Recherche Opérationelle: Recherches Locales, Réseaux Néuronaux et Satisfaction de Contraintes.* Hermes Science Publications, 2002. ISBN 2746204622.

V. T'Kindt and J.-C Billaut. *Multicriteria Scheduling: Theory, Models And Algorithms.* Springer-Verlag Berlin and Heidelberg GmbH & Co. K, 2nd ed. edition, 2006. ISBN 3540282300.

S. Webster and K. R. Baker. Scheduling groups of jobs on a single machine. *Operations Research*, 43(4):692–703, 1995.

S. T. Webster. The complexity of scheduling job families about a common due date. *Operations Research Letters*, 20(2):65–74, 1997. ISSN 0167-6377.

L. M. Wein. Scheduling semiconductor wafer fabrication. *IEEE Transactions on Semiconductor Manufacturing*, 1(3):115–130, 1988. ISSN 0894-6507.

E. D. Wikum, D. C. Llewellyn, and G. L. Nemhauser. One-machine generalized precedence constrained scheduling problems. *Operations Research Letters*, 16(2):87–99, 1994. ISSN 0167-6377.

F. Zaugg. Single wafer management for non productive wafers NPW. In *IEEE/SEMI Advanced Semiconductor Manufacturing Conference 2008*, pages 191–195, 2008. ISBN 978-1-4244-1964-7.

H. Zhou, Z. Li, and X. Wu. Scheduling unrelated parallel machine to minimize total weighted tardiness using ant colony optimization. In *IEEE International Conference on Automation and Logistics 2007*, pages 132–136, 2007.

École Nationale Supérieure des Mines
de Saint-Étienne

NNT : 2012 EMSE 0650

Ali OBEID

SCHEDULING AND ADVANCED PROCESS CONTROL IN SEMICONDUCTOR MANUFACTURING

Specialty: Industrial Engineering

Keywords: Scheduling, Advanced Process Control, Parallel machines, Semiconductor manufacturing, Time constraints, Health indicators, Integer Linear Programming, Heuristics

Abstract:

In this thesis, we study various possibilities of integrating scheduling decisions with Advanced Process Control (APC) systems in Semiconductor Manufacturing. Important questions are opened regarding the benefits of integrating scheduling and APC. In particular, two new scheduling problems are defined: Problem of Scheduling with Time Constraints (PTC) and Problem of Scheduling with Equipment Health Factors (PEHF). PTC and PEHF have multi-criteria objective functions. Mathematical models and various heuristics are proposed to solve these problems, and are validated with numerous experimentations.
PTC aims at scheduling lots of different families on non-identical parallel machines with setup times. Machine are not identical since they cannot (are not qualified to) process all lot families. Time constraints are related to the needs of APC control loops to be regularly fed with information. The qualification of a lot family on a machine is lost if the time between two consecutive lots of the family is larger than a given threshold. The objective is to minimize the sum of the lot completion times and the losses in machine qualifications.
PEHF is an extension of PTC which aims at integrating an indicator modeling the state of the machine (EHF, Equipment Health Factor). The goal is to minimize the sum of the lot completion times, the losses in machine qualifications and to maximize the yield. The yield of an operation is defined as a function of the EHF of the machine and of the job criticality.

École Nationale Supérieure des Mines
de Saint-Étienne

NNT : 2012 EMSE 0650

Ali OBEID

ORDONNANCEMENT ET CONTRÔLE AVANCÉ DES PROCÉDÉS EN FABRICATION DES SEMI-CONDUCTEURS

Spécialité : Génie Industriel

Mots clefs : Ordonnancement, Contrôle avancé des procédés, Machines parallèles, Fabrication de semi-conducteurs, Contraintes de temps, Indices de santé, Programmation linéaire en nombres entiers, Heuristiques

Résumé :

Dans cette thèse, nous étudions différentes possibilités d'intégration des décisions d'ordonnancement avec les systèmes de contrôle avancé des procédés (APC) en fabrication de semi-conducteurs. En particulier, nous définissons deux nouveaux problèmes d'ordonnancement : problème d'ordonnancement avec contraintes de temps (PTC) et problème d'ordonnancement avec prise en compte de l'état de santé des équipements (PEHF). PTC et PEHF ont des fonctions objectives multicritères. Des modèles mathématiques et différentes heuristiques sont proposés pour résoudre ces problèmes, et validés avec de nombreuses expérimentations.

PTC est un problème d'ordonnancement de lots de différentes familles sur des machines parallèles non identiques avec des temps de setup. Les machines sont non identiques car elles ne peuvent pas traiter (être qualifiées pour) toutes les familles de lots. Les contraintes de temps sont liées à la nécessité d'alimenter régulièrement par des informations sur les lots les boucles de régulation en contrôle avancé des procédés. La qualification d'une famille de lots sur une machines est perdue si le temps entre deux lots consécutifs de la famille dépasse un seuil de temps donné. L'objectif est de minimiser la somme des dates de fin des lots et les pertes de qualification des machines.

PEHF est une extension de PTC, qui vise à intégrer un indicateur modélisant l'état de la machine (EHF, *Equipment Health factor*). L'objectif est de minimiser la somme des dates de fin des lots, les pertes de qualification des machines et de maximiser le rendement. Le rendement d'une opération est défini comme une fonction de l'EHF de la machine et de la criticité du lot.