# Extraction de règles d'ordonnancement : Aide au paramétrage d'un progiciel d'ordonnancement.

Brigitte Chebel-Morello, Pierre Baptiste, Emmanuel Lereno

# Extraction de règles d'ordonnancement : Aide au Paramétrage d'un Progiciel d'ordonnancement

**Brigitte Chebel_Morello*— Pierre Baptiste**— Emmanuel Lereno**

**Laboratoire d'Automatique de Besançon*
*CNRS UMR 6596, ENSMM - UFC*
*25, rue Alain Savary 25000 BESANCON - France*
*bmorello@ens2m.fr*

***Département de Mathématiques et de génie industriel*
*Ecole Polytechnique*
*2500, chemin de Polytechnique H3T 1J4*
*pierre.baptiste@polymtl.ca*

RÉSUMÉ : *Les fluctuations du marché complexifient les conditions de pilotage d'un atelier de production et nécessitent différentes heuristiques d'ordonnancement adaptées au contexte. La principale difficulté réside alors dans le choix de l'heuristique à appliquer. Nous proposons une méthode d'aide à l'ordonnancement, et plus spécifiquement d'aide au paramétrage d'un logiciel d'ordonnancement s'appuyant sur un système d'apprentissage, capable d'extraire de la connaissance. On resituera ce système dans un processus d'E.C.D. (Extraction de Connaissances à Partir de Données, qui présente une première étape spécifique à notre problème et utilise les capacités de simulation d'un logiciel d'ordonnancement du marché. Ces logiciels constituent parfois le seul outil d'aide à l'ordonnancement dans les PME/PMI, utilisés en général en deçà de leur potentiel. Nous proposerons une application permettant de les utiliser au mieux.*

ABSTRACT: *Conditions complexity of management workshop of production manufacturing increases and requires heuristic algorithm adapted to the context. The principal difficulty then lies in the choice of heuristic algorithm to apply. We propose a help method to schedule, and more specifically a parameter setting up's help of an industrial scheduling software, being based on machine learning system able to extract knowledge from data. An inductive learning based on examples system is developed and replaced in a process of ECD (Extraction of Knowledge starting from Data. The first step of this process is specific to our problem and uses in this case the capacities of simulation of a market software of scheduling.*

MOTS-CLÉS: *ordonnancement, paramétrage, règles de connaissance, simulation, aide à la décision, filtrage*

KEYWORDS : *scheduling, knowledge rules, simulation, filtering*

## 1. Nécessité d'une connaissance minimale dans un atelier de production

### 1.1. *Introduction*

La recherche systématique d'une plus grande productivité, la prise en compte de plus en plus de critères liés aux ressources soit machines (indisponibilité des machines pour cause de maintenance ou de défaillances), soit humaines (avec la gestion du personnel et leur indisponibilité éventuelle en cas de maladie), le contrôle et la gestion fine des opérations élémentaires dans un atelier de production deviennent fondamentales. Les systèmes de production sont par conséquent de plus en plus complexes et l'ordonnancement de tels ateliers est un problème qui ne peut être résolu par les seuls algorithmes optimaux. Ordonnancer consiste à affecter à des tâches des ressources et un espace temporel d'exécution, en prenant soin de respecter un ensemble de contraintes (Blasewicz *et al.,* 1993). D'un point de vue concret, il s'agit de réguler le passage de chaque produit (Ordre de Fabrication) sur un ensemble de postes de travail en intégrant les contraintes de date (respect des délais, opérations de maintenance), d'ordre (gamme) ou de ressources dites secondaires (opérateurs, machines). Il s'agit en fait d'un problème d'optimisation combinatoire, où il faut choisir une bonne solution, voire une solution optimale, par rapport à un critère ou à un ensemble de critères d'évaluation. En effet le plus souvent la résolution de tels problèmes est obtenue par simulation de file d'attente. C'est une solution approchée qui est la moins gourmande en temps de calcul et est proposée par les logiciels du marché.

En raison des fluctuations de la demande du marché, les conditions de l'atelier peuvent évoluer d'une semaine à l'autre et demander l'utilisation de différents types d'heuristiques. La principale difficulté réside alors dans le choix de la règle de gestion à appliquer. Ainsi, chaque heuristique peut se révéler intéressante dans un certain nombre de cas particulier et totalement inadéquat pour d'autres. Dés lors, il s'avère très difficile de choisir avec certitude laquelle utiliser face à un problème d'ordonnancement précis.

Nous proposons de solutionner ce problème par l'utilisation d'un système d'apprentissage répondant à la variabilité des demandes du marché. Plusieurs auteurs se sont intéressés à ce type de problème afin de déduire des règles de décision permettant de faire ce choix. Nous passons en revue à la section suivante les différents travaux faits sur ces méthodes. Une des spécificités de notre système d'apprentissage est de permettre non seulement une meilleure utilisation des ressources matérielles mais également humaines dans l'entreprise.

A la lumière de nos travaux dans le domaine, nous présentons en deuxième section une méthodologie permettant d'apprendre les spécificités de l'atelier grâce à une phase de résolution de problèmes nécessitant une étape de simulation. Cette étape permettra de générer une base de données technique liée à l'ordonnancement de l'atelier, qui sera exploitée ensuite par un processus d'extraction des connaissances à partir des données (E.C.D.)

La connaissance extraite de la base de données technique peut être exploitée de différentes façons (Michaut et *al.,*1998) (Harrath et *al.,*2003). Nous l'orientons dans cette étude vers le paramétrage d'un logiciel commercial d'ordonnancement.
Ce système d'apprentissage sera appliqué au paramétrage de logiciel d'ordonnancement et nous montrons en section 1.3 cette nécessité qui se fait sentir à l'exploitation de ce progiciel
Enfin, le troisième paragraphe décrira une application concernant un problème d'ordonnancement d'un atelier flow shop hybride de production de gourmettes en or.

### 1.2. *Etat de l'art*

Maints auteurs se sont penchés sur la difficulté à choisir l'heuristique la plus appropriée au contexte étudié. Entre autre Kassou a proposé un modèle orienté objet basé sur des méthodes de voisinage hybride permettant de tester différentes heuristiques (Kassou et *al*., 1998). D'autres auteurs ont proposé de mettre en place des systèmes d'apprentissage accédant ainsi à la connaissance sous forme de règles permettant de faire ce choix. (Geneste et *al.,* 1996) (Michaut et *al.,* 1998) (Lereno et *al.,* 2001)
Les 2 types d'apprentissage existant ont été proposés à savoir l'apprentissage déductif, comme les systèmes expert basés sur la connaissance experte, traduite en règles qui appliquées par un moteur d'inférence permet un raisonnement déductif (Geneste et *al.,*1997), et l'apprentissage inductif qui extrait la connaissance à partir des données simulées (Chebel-Morello et *al.,*2001). N'ayant pas de connaissance *a priori* sur notre atelier, nos travaux se sont orientés vers l'apprentissage inductif à partir d'exemples, que l'on replacera dans un processus complet d'extraction des connaissances à partir des données.
Cette connaissance extraite permettra également de faire des liens entre toutes ces données et de paramétrer si besoin est un logiciel commercial d'ordonnancement. En effet l'utilisation d'un logiciel commercial nécessite une connaissance minimale de l'atelier et requiert de l'aide pour une utilisation à la hauteur des services proposés. Ces logiciels font intervenir un nombre conséquent de paramètres, dont l'ajustement a une influence non négligeable sur la qualité de l'ordonnancement. Nous proposons donc d'élaborer un système d'apprentissage dédié à ce paramétrage, après avoir montré cette nécessité à la section suivante.
L'automatisation du réglage des paramètres intervenant en entrée d'un logiciel d'ordonnancement par rapport à un cas industriel a intéressé un certain nombre d'auteurs. Des modèles de simulation ont été proposés en tenant compte des détails et des contraintes permettant d'estimer avec une bonne précision l'effet de paramètres par rapport à des critères de performances, sur un ensemble de données. Paris a proposé dans le cadre d'une configuration multi produit dans un système Kanban, une optimisation par des algorithmes évolutionnaires distribués (Paris et *al.,*2001), alors que Huyet couple l'algorithme évolutionnaire avec un système d'apprentissage (Huyet et *al.,*2003) comme (Harrath et *al.,* 2003). Talbi, s'est interessé à sélectionner parmi un grand nombre de paramètres, ceux qui ont une

influence sur la qualité de l'ordonnancement, par des algorithmes évolutifs. (Talbi et *al.,*2003) (Talbi et *al.,*2004)
Geneste a proposé un système expert, Michaut un système d'apprentissage inductif appliqué au choix de la meilleure heuristique à appliquer sur une configuration d'atelier. Nos travaux, sont une extension de (Chebel-Morello et *al.,*2001), appliquée au paramétrage d'un logiciel d'ordonnancement et tenant compte de nombreux paramètres de réglage à savoir l'utilisation des ressources principales que sont les machines, les opérateurs qui deviennent de plus en plus souvent les ressources critiques avec les calendriers opératoires associés, les heuristiques d'ordonnancement, et l'ordonnancement lui même.
Cette approche prend appui sur un système d'apprentissage qui génère une base de connaissances sur le domaine considéré, que nous exploitons dans le système d'aide à la décision.

### 1.3. *Nécessité d'un paramétrage de logiciel d'ordonnancement*

Le marché se partage aujourd'hui entre plusieurs distributeurs de logiciels, soit en satellite de Progiciel de Gestion Intégrée PGI en Français ou ERP « Entreprise Resource Planning » soit en tant que logiciel indépendant, qui ont de nombreux points communs. Des points faibles qu'offrent ces logiciels commerciaux nous en retiendrons deux :

– ils partent tous de calendriers d'ouverture des ressources aussi bien machines qu'opérateurs, qu'ils considèrent comme des données d'entrée alors que les utilisateurs aimeraient avoir ces données comme des résultats attendus des logiciels, pour avoir une première base de travail.

–ils disposent de nombreux paramètres réglables qui, dans l'absolu peuvent se changer en temps réel, mais qui sont le plus souvent positionnés une fois pour toute vu le peu de maîtrise des utilisateurs

Les utilisateurs doivent tâtonner pour pouvoir modifier les calendriers, et beaucoup plus rarement les paramètres réglables en simulant des ordonnancements qu'ils analysent. Si les résultats escomptés ne sont pas satisfaisants, les utilisateurs itèrent.
Afin d'éviter ce tâtonnement et permettre l'utilisation de ces logiciels au mieux de leur possibilité, nous proposons d'effectuer un apprentissage des paramètres à régler, sur des données spécifiques à l'atelier et de fournir ainsi une connaissance adaptée à l'atelier sous forme de règles de paramétrage de ces logiciels.
La méthode adoptée pour cet apprentissage suit la démarche intuitive mais manuelle des utilisateurs, tout en lui donnant une toute autre ampleur par la génération d'une base de données technique, qui sera analysée par des méthodes d'apprentissage à partir d'exemples.

## 2. Démarche proposée

### 2.1. *Introduction*

Afin d'effectuer un apprentissage des paramètres à régler, nous proposons d'élaborer une base de données techniques, et d'extraire de la connaissance sur les règles d'ordonnancement de l'atelier, grâce à un processus d'E.C.D.
L'E.C.D. ou (Knowledge Discovery in Databases) est un "processus non trivial d'identification de structures inconnues, valides et potentiellement exploitables dans les bases de données" (Fayyad et al., 1996). Elle se réfère à une démarche complète d'exploitation des données que l'on peut résumer (Liu et *al.* 1998) en quatre phases distinctes :

– la collecte de données qui consiste à homogénéiser les données collectées sur différentes bases de données et à les compléter

– une étape de prétraitement nettoyant les données et les filtrant

– une l'étape de fouille de données, qui mettra en évidence la structure sous jacente des données et les proposera dans certains cas sous forme de règles de décision

– une mise en forme de ces règles.

Dans le cas présent la collecte des données sera spécifique au problème de paramétrage, et nécessitera une phase de simulation couplée à une phase de construction d'ordonnancement notamment faite à l'aide du progiciel commercial , suivi d'une phase d'évaluation permettant de sélectionner l'ordonnancement le mieux adapté.
La représentation du problème sera faite en sorte que la base de données résultante puisse être exploitée par un processus d'E.C.D., et fournir ainsi des règles compréhensibles par un utilisateur néophyte du logiciel d'ordonnancement.

Soit $\Pi$ la population étudiée
$\Omega$ L'échantillon de n exemples observés, ou instances $\Omega=\{\omega_1, \omega_2,\ldots,\omega_n\}$
Y l'ensemble de r variables observées $Y =\{ y_1,\ldots,y_k,\ldots,y_r \}$ défini sur $\Pi$.
C la variable qualitative prenant les valeurs associées aux classes possibles.

Il existe deux types de méthodes en apprentissage inductif : l'apprentissage de concept à partir d'observations et à partir d'exemples.
Une observation en apprentissage de concept à partir d'observations ou apprentissage non supervisé est définie par un ensemble de variables Y
Un exemple en apprentissage de concepts à partir d'exemples ou apprentissage dit supervisé est défini par un ensemble d'observations déjà pré classées par un professeur en plusieurs classes (Y, C) (Michalski et *al.,*1978)
Nous travaillerons sur des exemples décrivant la configuration d'un atelier associé à des classes de résolution de problèmes. Le pré classement se fera à l'aide du solveur de problèmes du progiciel, dans l'objectif de pallier le manque d'expertise des utilisateurs.

### 2.2. *Résolution de problèmes ou collecte de données*

L'élaboration de notre population d'apprentissage via l'étape de collecte des données, consistera à générer les configurations d'un atelier et de les ranger en classes de résolution de problèmes. La collecte de données, sera composée de trois parties et permettra d'acquérir des exemples venant alimenter une base de données technique.

I°- génération automatique de différentes configurations d'atelier décrites par des caractéristiques

II°- Etiquetage de chaque configuration par une classe de résolution de problèmes soit par :

– un expert (ce qui est généralement fait lorsque l'expertise existe)

– simulation d'un ordonnancement nécessitant pour chaque configuration :

a - une étape *de construction d'ordonnancement* par différentes heuristiques

Réalisée par le logiciel d'ordonnancement

b - une étape *d'évaluation des heuristiques* suivants différents critères

c – une étape *de sélection multicritères* de la meilleure heuristique

III°- sauvegarde dans la base de données technique BDT de l'exemple ainsi formé.

La base de données est ainsi formée d'exemples, où chaque exemple est composé d'une configuration de l'atelier (ses ressources machines et humaines), associée à la classe d'heuristique la plus adaptée à sa résolution.

IV- la fouille de données et l'étape de filtrage de variables et d'exemples.

Cette base BDT sera soumise aux différentes étapes du processus d'E.C..D.

En l'occurrence la discrétisation des données numériques en données symboliques afin de les homogénéiser et de les généraliser, une étape de filtrage de données qui permet de réduire la BDT aux exemples les plus pertinents ( nous ne développerons pas cette étape dans ce papier ) et une étape de fouille de données utilisant les arbres de décision intelligible.

### 2.3. *Fouille de données et extraction de règles de connaissance*

La population d'apprentissage ainsi obtenue dans la BDT, sera présentée à un mécanisme d'apprentissage que nous avons réalisé par un arbre de décision.
Les arbres de décision sont des systèmes d'apprentissage inductif à partir d'exemples et s'inscrivent dans l'E.C.D.. En effet, ils permettent de distinguer les structures sous-jacentes qui régissent les données et de construire des règles capables de classer les objets à partir d'un ensemble d'apprentissage constitué d'objets étiquetés (les classes sont connues).
A partir d'une population convenablement étiquetée. Le processus d'obtention d'un d'un arbre de décision (équivalent à un classifieur) peut être généralement résumé en

deux étapes principales. L'une des méthodes les répandues consiste à diviser l'ensemble d'apprentissage en deux sous-ensembles : un ensemble d'entraînement $\Omega_1$ (training set) et un ensemble test $\Omega_2$ (test set). Le premier de ces ensembles est utilisé afin d'induire un classifieur(l'arbre de décision) et le second sert à la validation du classifieur obtenu. L'algorithme d'induction choisi est exécuté jusqu'à ce que la précision de la classification pour l'ensemble $\Omega_1$ satisfasse un seuil choisi à l'avance. Puis, la seconde phase consiste à mesurer la performance du classifieur. Pour cela, l'ensemble $\Omega_2$ est alors utilisé comme donnée d'entrée du classifieur et on mesure sa performance en évaluant le nombre d'objets appartenant à $\Omega_2$ et correctement classés L'arbre de décision ainsi obtenu représente une base de règles. Une règle *si alors* est exprimée sous forme d'un chemin partant de la racine pour atteindre une feuille (nœud terminal) d'un arbre. Chaque nœud interne est associé à un attribut à tester et à autant de sous arbres que l'attribut comporte de valeurs. Chaque nœud terminal possède une classe assignée exprimant le résultat d'une règle de classification. L'idée de la construction d'arbres à partir d'un ensemble d'apprentissage a été proposée initialement dans (Hunt et *al.,*1996). De nombreux travaux de recherche ont été menés dans le domaine, nous citerons parmi les plus connus CART (Breiman et *al.,* 1984) et C4.5 (Quinlan et *al.,*1983) (Quinlan et *al.,*1993)

## 3. Mise en œuvre de la démarche avec élaboration d'ordonnancement par Preactor

Afin de réaliser la partie simulation de ce travail, nous avons utilisé un logiciel d'ordonnancement couramment répandu dans les entreprises, le logiciel *Preactor*. Ce type de logiciel accepte en entrée de nombreux paramètres décrivant d'une part l'atelier de production et d'autre part les ordres de fabrication. Avant d'aller plus avant nous recensons les données manipulées dans un atelier

### 3.1. *Les données manipulées dans un atelier*

Dans un logiciel commercial, nous trouvons 3 types de données.

– les données d'entrée, les paramètres à étudier et les différents critères d'évaluation de ces paramètres.

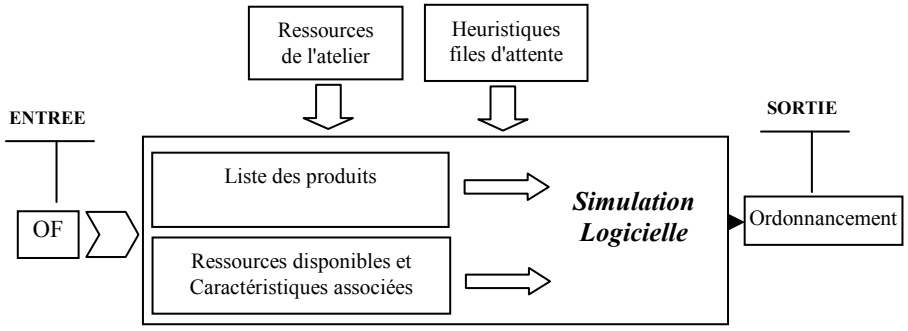


**Figure 1.** *Données manipulées dans l'atelier*

Les données d'entrée sont composées de données statiques et de données dynamiques.

– *Les Données statiques :*

Elles représentent l'état initial du système. Elles codifient la dénomination des produits susceptibles d'être fabriqués ainsi que la description des ressources présentes dans l'atelier

– *Les Données dynamiques comprennent :*

- les OF ordres de fabrication qui fluctuent à la semaine suivant le marché et les commandes ; les degrés de liberté telles que les ressources humaines de l'atelier affectées aux machines, qu'on prendra comme paramètres qu'il est possible de faire varier pour évaluer un ordonnancement

- Et les différents critères d'évaluation à travers une fonction coût, dépendant du poids que l'on donne par exemple à la charge des machines, à la date de livraison etc...

### 3.2. *Collecte de données ou Génération de la base de données techniques*

I°- génération automatique des différentes configurations

Préactor est un logiciel dédié à la simulation d'ordonnancement d'ateliers de production. Où l'on doit décrire le fonctionnement de l'atelier en définissant les données statiques, et les données dynamiques.

*a- Description de l'atelier*

- modélisation de l'entreprise dans le progiciel « Préactor » :

On doit déclarer donc dans Preactor la configuration de l'atelier avec ses postes de travail, les calendriers, le type de produit fabriqué, donner la définition des produits en spécifiant les articles nomenclatures opérations et gammes.

*b- Les données dynamiques les degrés de libertés :*

On doit définir également les Ordres de Fabrication, nombre d'O.F., les quantités d'article les délais. C'est à ce stade que la phase de simulation intervient. Les données que nous ferons plus particulièrement évoluer sont les ordres de fabrication, les ressources humaines, les horaires de travail et enfin la gestion de files d'attente.

Pour cela, nous avons fait appel à des outils de programmation en langage C++, et généré ainsi un ensemble d'exemples représentant une partie des données d'entrée dynamiques du système, à savoir :

- un ensemble de semaines auxquelles nous associons des OF en prenant soin, en fonction de l'atelier considéré, de couvrir au mieux l'espace des situations auxquelles l'organe décisionnel de l'atelier doit faire face.

II°- Etiquetage de chaque configuration

*a- Elaboration d'ordonnancement par l'utilisation de progiciel*

Tout type d'heuristique proposé par Preactor peut être testé :

*b- Evaluation d'heuristiques*

Différents critères pour l'analyse des différents ordonnancements obtenus peuvent être sélectionnés. L'évaluation de chaque ordonnancement peut être faite en terme d'heures payées, d'encours ou de stock moyen, de la durée totale d'ordonnancement Cmax[1], de la charge des machines, du volume global de la charge,

*c- Sélection d'heuristiques*

La combinaison de deux critères, le coût financier d'une semaine de production et le Cmax associé à l'ordonnancement a permis d'évaluer la performance de chaque l'ordonnancement et de sélectionner les heuristiques et les configurations les mieux adaptées au bon fonctionnement de l'atelier.

III°- Sauvegarde de la BDT

Ainsi le paramétrage des variables du système pour le meilleur résultat obtenu est ajouté à la base de données technique. Cette information sur l'ajustement des degrés de liberté est l'étiquette qui correspond à la classe de l'exemple (semaine de production) correspondant.

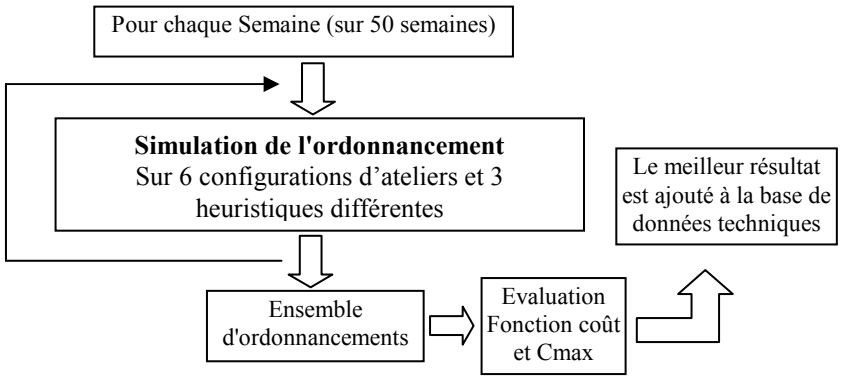


**Figure 2.** *Constitution de la Base de données technique du système*

De ce fait, il peut être nécessaire de déterminer chaque semaine, pour un même atelier, l'ajustement d'heuristiques ou de ressources liées à l'atelier qui est susceptible de fournir les meilleurs résultats.

[1] Cmax = Makespan ou durée totale d'ordonnancement

### 3.3. *Les structures classificatoires*

C'est dans cette optique que nous proposons un apprentissage inductif conceptuel à partir d'exemples (Diday *et al.,*1989). Ainsi, nous recherchons les descriptions les plus générales possibles expliquant les données d'entrée, dans le but de prédire un comportement du système et de décider quel ajustement des degrés de liberté correspond le mieux à une situation nouvelle.

Nous avons utilisé dans notre étude l'algorithme C4.5 (Quinlan., 1983) Celui-ci nous permet d'extraire un ensemble de règles à partir des données extraites de la base de données techniques que nous avons constituée. Ces règles peuvent ainsi être utilisées par le système pour affecter une semaine de production à l'une des classes obtenues lors du processus d'apprentissage.

## 4. Application à un Problème d'Ordonnancement

Nous considérons le cas d'une entreprise désirant perfectionner son système d'ordonnancement d'atelier. Cette société utilise l'or comme matière première et fabrique des gourmettes et des chaînes pour homme, femme et enfant. Chaque semaine les "donneurs d'ordres" contactent cette entreprise et passent leurs commandes. Les produits sont ainsi fabriqués durant la semaine en considérant qu'aucun or ne doit rester dans les ateliers de fabrication durant le week-end afin de réaliser des économies sur les coûts de stockage.

### 4.1. *L'atelier de fabrication : un flow shop hybride*

#### 4.1.1. Le Procédé de Fabrication

Le procédé de fabrication comporte six phases auxquelles une ou plusieurs machines peuvent être associées afin de réaliser la tâche demandée. La Figure 3 présente succinctement le processus de fabrication d'un produit (chaîne ou gourmette) à travers ces six étapes.

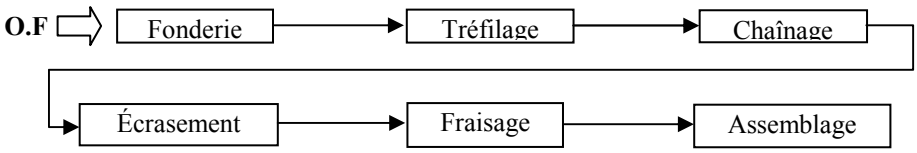


**Figure 3 :** *Les différentes étapes de fabrication*

#### 4.1.2. Le Fonctionnement de l'Atelier

Nous disposons dans l'atelier d'un poste (machine) par tâche à réaliser excepté dans le cas du tréfilage où deux postes sont mis en place. En effet, la charge de travail lors de la seconde étape de fabrication justifie l'utilisation de deux

machines. La production est linéaire, signifiant que le flux de produits est unique, en effet les mêmes transformations sont réalisées sur tous les produits.

4.1.3. Caractéristiques des Produits

L'atelier produit des chaînes en or et propose à ses clients 30 variations possibles du produit. En effet, les maillons de la chaîne auront des diamètres et une longueur plus ou moins importants suivant la personne à qui elle est destinée (hommes, femmes, enfants). La gourmette est considérée comme un cas particulier d'une chaîne, à laquelle est adjointe une plaquette. A chaque type de chaîne va donc être associé un temps de travail nécessaire à chaque étape du procédé de fabrication. Le produit $P_i$, $i \in [1,2,.., 30]$ sera caractérisé par le vecteur temps suivant :

$P_i = (t_i^1, t_i^2, t_i^3, t_i^4, t_i^5, t_i^6)$ avec $t_i^k$ , $k \in [1,2,..., 6]$ les temps associées aux machines $M_1$, $M_2$, …, $M_6$.

Ces temps seront calculés, dans notre générateur de cas, en fonction de la description du processus de fabrication donné précédemment.

### 4.2. *Les données manipulées : la Génération des Commandes*

Nous considérons que les donneurs d'ordre passent leur commande de manière à ce que la production puisse démarrer chaque lundi matin. Un OF est considéré dans notre cas comme un couple ($P_i$,$Q_i$), $Q_i$ déterminant la quantité de produit $P_i$ à fabriquer. Soit $N_{of}$ le nombre d'OF pour une semaine donnée. Le générateur peut fonctionner suivant trois modes différents :

– **mode 1** : génération de semaines dites "classiques" où $N_{of}$ et $Q_i$ varient uniformément d'une semaine à l'autre (la demande porte sur l'ensemble des produits dans des quantités variables),

– **mode 2** : génération de semaines où Nof est petit avec Qi grand (ce cas de figure peut correspondre à quelques modèles particulièrement demandés et qu'il faut rapidement produire en grande quantité),

– **mode 3** : génération de semaines où $N_{of}$ est grand et $Q_i$ petit (cas où la demande est uniforme sur l'ensemble des produits mais dans de petites quantités).

Nous avons ainsi généré un grand nombre de semaines de production réparties comme suit : 60% sur le mode 1, 20% sur le mode 2 et 20% sur le mode 3. Parmi ces semaines de production, 60% ont été aléatoirement sélectionnées pour la phase d'entraînement du système d'apprentissage, les 40% restants servant à l'évaluation de la qualité du classifieur obtenu.

### 4.3 *Collecte de données : Simulation et Evaluation de l'Ordonnancement*

Nous désirons étudier la charge de travail des machines de l'atelier durant la semaine. Nous avons retenons les variables descriptives suivantes.

- **Charge globale** : Charge totale de travail pour une semaine,
- **Nombre d'OF** : Nombre d'ordres de fabrication,
- **Charge M1, …, Charge M6** : Charge de travail pour chaque machine.

Un échantillon des valeurs numériques associées à ces données est présenté dans le Tableau 1. Nous disposons de 80 semaines de commandes que nous avons réparties de la manière suivante :

– 50 semaines dédiées à la phase d'apprentissage (données d'environ un an),

– 30 semaines dédiées à la phase de test (horizon prévisionnel d'environ 7 mois).

Remarquons que nous avons pris soin, dans chacun des deux ensembles, de conserver une distribution homogène entre les semaines générées selon les modes 1, 2 ou 3.

| | *y1* | *y2* | *y3* | *y4* | *y5* | *y6* | *y7* | *y8* |
|---|---|---|---|---|---|---|---|---|
| | **Charge globale** | **Nombre d'OF** | **Charge M1** | **Charge M2** | **Charge M3** | **Charge M4** | **Charge M5** | **Charge M6** |
| **Semaine 1** | 146,43 | 16 | 24,41 | 36,11 | 18,72 | 22,73 | 28,00 | 16,46 |
| **Semaine 2** | 175,97 | 18 | 35,34 | 35,57 | 19,73 | 27,99 | 39,00 | 18,33 |
| **Semaine 3** | 134,80 | 12 | 27,62 | 28,22 | 16,43 | 21,15 | 27,00 | 14,38 |
| **Semaine 4** | 155,12 | 14 | 26,31 | 38,55 | 18,78 | 24,56 | 29,00 | 17,92 |
| **Semaine 5** | 130,11 | 17 | 20,81 | 31,63 | 16,62 | 19,68 | 27,00 | 14,38 |
| **Semaine 6** | 132,23 | 20 | 26,46 | 28,63 | 15,90 | 19,28 | 28,00 | 13,96 |
| **Semaine 7** | 205,84 | 21 | 38,21 | 46,95 | 25,25 | 31,14 | 42,00 | 22,29 |
| **Semaine 8** | 127,00 | 11 | 17,56 | 34,33 | 16,68 | 20,43 | 23,00 | 15,00 |
| **Semaine 9** | 209,30 | 21 | 37,86 | 49,77 | 26,77 | 32,36 | 39,00 | 23,54 |
| **Semaine 10** | 131,39 | 13 | 19,01 | 35,85 | 18,48 | 19,63 | 23,00 | 15,42 |
| **Semaine 11** | 205,91 | 24 | 33,05 | 52,57 | 28,58 | 31,16 | 37,00 | 23,54 |
| **…..** | **…..** | **…..** | **…..** | **…..** | **…..** | **…..** | **…..** | **…..** |

**Tableau 1.** *Attributs sélectionnés pour l'apprentissage (valeurs continues)*

Pour chaque semaine de travail, nous avons fait évoluer le nombre, la disposition des opérateurs dans l'atelier et les horaires de travail associés. Huit configurations pour une semaine de travail ont été définies. Le Tableau 2 montrent la répartition du personnel et les horaires de travail associés suivant deux types de configuration ; la classique correspondant aux horaires 8h00-12h00 / 14h00-18h et la configuration 2/8. Nous tenons également compte de plusieurs distributions de personnel sur les machines pour une configuration donnée. Le Tableau 3 nous montre les deux possibilités envisagées pour la configuration 5.

Notons que les résultats obtenus lorsque Preactor évalue l'ordonnancement diffèrent pour chacune d'entre elles.

Remarquons que dans l'exemple considéré, la machine M2.2 (tréfileuse 2) n'intervient pas. Ce n'est, bien entendu, pas toujours le cas.

| | Semaines en 2/8 | | | | | Semaines "classiques" | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Horaire | Nombre d'opérateurs | | | | | Nombre d'opérateurs | | | |
| | | *cfg5* | *cfg6* | *cfg7* | *cfg8* | Horaire | *cfg1* | *cfg2* | *cfg3* | *cfg4* |
| **lundi** | 04h – 12h | 2 | 3 | 3 | 3 | 08h – 12h | 3 | 4 | 5 | 6 |
| | 12h– 20h | 2 | 3 | 3 | 3 | 14h – 18h | 3 | 4 | 5 | 6 |
| **mardi** | 04h – 12h | 2 | 3 | 3 | 3 | 08h– 12h | 3 | 4 | 5 | 6 |
| | 12h – 20h | 2 | 3 | 3 | 3 | 14h – 18h | 3 | 4 | 5 | 6 |
| **mercredi** | 04h – 12h | 2 | 3 | 3 | 3 | 08h – 12h | 3 | 4 | 5 | 6 |
| | 12h – 20h | 2 | 2 | 1 | 3 | 14h – 18h | 3 | 4 | 5 | 6 |
| | *08h– 12h* | - | 1 | 2 | - | | | | | |
| | *14h – 18h* | - | 1 | 2 | - | | | | | |
| **jeudi** | 04h – 12h | 2 | 3 | 3 | 3 | 08h – 12h | 3 | 4 | 5 | 6 |
| | 12h – 20h | 2 | 2 | 2 | 3 | 14h – 18h | 3 | 4 | 5 | 6 |
| | *08h– 12h* | - | 1 | 1 | - | | | | | |
| | *14h– 18h* | - | 1 | 1 | - | | | | | |
| **vendredi** | 04h – 12h | 2 | 2 | 3 | 3 | 08h – 12h | 3 | 4 | 5 | 6 |
| | 12h – 20h | 2 | 3 | 3 | 3 | 14h – 18h | 3 | 4 | 5 | 6 |
| | *08h – 12h* | - | 1 | - | - | | | | | |
| | *14h – 18h* | - | 1 | - | - | | | | | |
| **samedi** | 04h – 12h | 2 | 3 | 3 | 3 | 08h – 12h | 3 | 4 | 5 | 6 |

**Tableau 2.** *Les différentes configurations dans l'atelier*

| | Lundi | | Mardi | | Mercredi | | Jeudi | | Vendredi | | Samedi | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 8h00 - 12h00 | 12h00 - 20h00 | 8h00 - 12h00 | 12h00 - 20h00 | 8h00 - 12h00 | 12h00 - 20h00 | 8h00 - 12h00 | 12h00 - 20h00 | 8h00 - 12h00 | 12h00 - 20h00 | 8h00 - 12h00 | |
| Fonderie | **1** | **1** | **1** | **1** | | | | | | | | - |
| Tréfileuse 1 | **1** | **1** | | | **1** | **1** | **1** | | | | | - |
| Chaînage | | | **1** | **1** | | | | **1** | | | | - |
| Aplatir | | | | | **1** | **1** | | | **1** | | | - |
| Fraisage | | | | | | | **1** | **1** | | **1** | **1** | - |
| Assemblage | | | | | | | | | **1** | **1** | **1** | - |

| | Lundi | | Mardi | | Mercredi | | Jeudi | | Vendredi | | Samedi | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 8h00 - 12h00 | 12h00 - 20h00 | 8h00 - 12h00 | 12h00 - 20h00 | 8h00 - 12h00 | 12h00 - 20h00 | 8h00 - 12h00 | 12h00 - 20h00 | 8h00 - 12h00 | 12h00 - 20h00 | 8h00 - 12h00 | |
| Fonderie | **1** | **1** | **1** | **1** | | | | | | | | - |
| Tréfileuse 1 | **1** | **1** | **1** | **1** | **1** | | | | | | | - |
| Chaînage | | | | | **1** | **1** | **1** | | | | | - |
| Aplatir | | | | | | **1** | | **1** | **1** | | | - |
| Fraisage | | | | | | | **1** | **1** | | **1** | **1** | - |
| Assemblage | | | | | | | | | **1** | **1** | **1** | - |

**Tableau 3.** *Solutions envisagées relatives à la répartition par poste de travail pour la configuration*

Nous disposons, à ce stade, de l'ensemble des données de base nécessaires à la constitution du jeu d'essais. Toutefois il faut mettre en place une fonction permettant d'évaluer le coût d'un ordonnancement puis d'en déduire le niveau de performance.

II Etiquetage de chaque configuration par une classe de résolution de problèmes

a- étape de résolution de problèmes.

Après expérimentation, nous avons conserver pour l'apprentissage les heuristiques donnant des résultats discriminant à la semaine : SPT, LAWR et une troisième heuristique combinant SPT et l'algorithme de Johnson.

– l'heuristique **LPT** (Longest Processing Time ou Temps Opératoire Maximum). Cette règle consiste à choisir dans la file d'attente d'une machine Mi l'ordre de fabrication dont le temps de passage sur Mi est le plus long,

– l'heuristique **SPT** (Short Processing Time ou Temps Opératoire Minimum). Cette règle consiste à choisir dans la file d'attente d'une machine Mi l'ordre de fabrication dont le temps de passage sur Mi est le plus court. Il est montré dans [Giard, 1988] que cette règle minimise l'attente moyenne des ordres de fabrication dans une file d'attente.

– l'heuristique **LAWR** (Least Actual Work Remaining), l'OF sélectionné est celui pour lequel la somme des $t_i^k$ restants est minimale

– l'heuristique de **Johnson** consiste, après avoir identifié les machines les plus chargées $M_i$, $M_j$ avec charge($M_i$)>charge($M_j$), à appliquer SPT sur $M_i$ et LPT sur $M_j$.

b- Fonction d évaluation du coût d'un ordonnancement

La tarification horaire

Afin de bâtir la fonction coût associé à l'ordonnancement, nous avons tout d'abord établi une tarification horaire prenant en compte les heures supplémentaires et les heures de nuit. Cette évaluation est dite "élémentaire", en effet nous sommes conscients que le coût d'un employé ne se limite pas à ces critères. Nous pensons qu'il s'agit toutefois d'une base de travail convenable.

Faisons le postulat que chaque employé travaille un nombre fixe d'heures par semaine, fixé pour l'exemple à 35 heures ce qui ne nuit en rien à la généralité de l'approche.

Soit C(h) le coût d'une heure de travail, on a :

- Coût d'une heure supplémentaire :

  - Si 1h ≤ nombre d'heures supplémentaires ≤ 8h : C(hs)=C(h) + 25%(C(h))
  - Si nombre d'heures supplémentaires ≥ 9h : C(hs)=C(h) + 50%(C(h))
- Coût d'une heure de nuit (de 21h00 à 6h00) : C(hn)=C(h) + 50%(C(h))

Dans le cas où l'employé doit effectuer des heures de nuit comptabilisées en plus comme des heures supplémentaires, les majorations s'additionnent. Pour exemple, le Tableau 5 présente deux tarifications horaires l'une relative à la configuration 4 et l'autre à la configuration 5 que nous avons définies auparavant.

| | Configuration4 | | | | Configuration 5 | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | Lundi / Jeudi | Vendredi | | Samedi | Lundi / Jeudi | | Vendredi | | Samedi |
| | 8h-12h 14h-18h | 8h-12h | 14h-18h | 8h-12h | 4h-12h | 12h-20h | 4h-12h | 12h-20h | 4h-12h |
| Nb personnes | 6 | 6 | | 6 | 2 | 2 | 2 | 2 | 2 |
| Nb jours | 4 | 1 | | | 4 | | 1 | | ½ |
| Nb heures "normales" | 8 | 7 | | 0 | 6 | 8 | 5 | 7 | 0 |
| Nb heures sup. (25%) | 0 | 1 | | 4 | 0 | 0 | 1 | 1 | 5 |
| Nb heures sup. (50%) | 0 | 0 | | 0 | 0 | 0 | 0 | 0 | 1 |
| Nb heures de nuit | 0 | 0 | | | 2 | 0 | 2 | 0 | 0 |
| Nb heures de nuit sup. | 0 | 0 | | | 0 | 0 | 0 | 0 | 2 |

**Tableau 5.** *Tableau d'évaluation du coût "élémentaire"des configurations 4 et 5*

Nous avons ainsi générés cinquante semaines fois huit configurations. Le Tableau 6 donne un aperçu partiel des données traitées. A chaque semaine, pour chaque configuration, les trois heuristiques sont appliquées et le Cmax de l'ordonnancement résultant est calculé ainsi que le coût financier de la semaine en fonction du nombre d'O.F.

| Full week | | | | | Mila+Chargée | | Cmax | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Semaines | TOTAL | M1 | ------ | M6 | 1ère | 2de | SPT | | LAWR | | SPT M1 + Johnso | | Best H | |
| 1 | **146,43** | 24,41 | ------ | 16,46 | 2 | 5 | 6565 | 1115 | 6508 | 1172 | 6565 | 1115 | 2 | 6508 |
| 2 | **175,97** | 35,34 | ------ | 18,33 | 5 | 2 | 7443 | 237 | 8399 | -719 | 7443 | 237 | 13 | 7443 |
| 3 | **134,80** | 27,62 | ------ | 14,38 | 2 | 1 | 6518 | 1162 | 6498 | 1182 | 6556 | 1124 | 2 | 6498 |
| 4 | **155,12** | 26,31 | ------ | 17,92 | 2 | 5 | 6668 | 1012 | 6679 | 1001 | 6675 | 1005 | 1 | 6668 |
| 5 | **130,11** | 20,81 | ------ | 14,38 | 2 | 5 | 6383 | 1297 | 6385 | 1295 | 6383 | 1297 | 13 | 6383 |
| 6 | **132,23** | 26,46 | ------ | 13,96 | 2 | 5 | 6358 | 1322 | 6358 | 1322 | 6348 | 1332 | 3 | 6348 |
| 7 | **205,84** | 38,21 | ------ | 22,29 | 2 | 5 | 7664 | 16 | 8537 | -857 | 7664 | 16 | 13 | 7664 |

Configuration 2 : 2 personnes 2/8 (version 2 du calendrier)

| Full week | | | | | Mila+Chargée | | Cmax | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Semaines | TOTAL | M1 | ------ | M6 | 1ère | 2de | SPT | | LAWR | | SPT M1 + Johnso | | Best H | | |
| 1 | **146,43** | 24,41 | ------ | 16,46 | 2 | 5 | 7505 | 175 | 7553 | 127 | 7568 | 112 | 123 | 7505 | |
| 2 | **175,97** | 35,34 | ------ | 18,33 | 5 | 2 | 8369 | -689 | 8759 | -1079 | 8362 | -682 | 13 | 8362 | X |
| 3 | **134,80** | 27,62 | ------ | 14,38 | 2 | 1 | 7536 | 144 | 7493 | 187 | 7501 | 179 | 123 | 7493 | |
| 4 | **155,12** | 26,31 | ------ | 17,92 | 2 | 5 | 7981 | -301 | 8262 | -582 | 8006 | -326 | 13 | 7981 | X |
| 5 | **130,11** | 20,81 | ------ | 14,38 | 2 | 5 | 7418 | 262 | 7493 | 187 | 7430 | 250 | 123 | 7418 | |

Configuration 3 : 3 personnes 2/8 dont 1 parfois 8-12/14-18 2

**Tableau 6.** *Echantillon des données traitées pour 2 configurations*

### 4.3 Génération d'une base de données techniques

4.3.1. Etiquetage de chaque configuration

Nous avons évalué, chaque ordonnancement obtenu en fonction du Cmax et du coût financier de la semaine de production. Nous avons ainsi dégagé trois catégories de semaines de production et le paramétrage d'atelier associé.
Dans le cas des calendriers de production, nous avons identifié trois configurations parmi l'ensemble des cas envisagés qui se révèlent être les solutions les plus efficaces
Remarquons que cela ne signifie pas que certaines autres configurations ne satisfaisaient pas les objectifs en termes de délais, toutefois le coût de leur mise en place étant plus important, elles ont été écartées de la solution finale. L'apprentissage par un algorithme d'induction de ces différentes configurations nous permettra de fournir l'information suivante au centre décisionnel :
"Etant donnés les ordres de fabrication de la semaine, la ***configuration*** $C_i$ est la meilleure solution en termes de coût et elle permet de terminer dans les délais impartis."

- Configuration 4 : 6 opérateurs, horaires de travail 8h-12h / 14h-18h,
- Configuration 5 (répartition 2) : 2 opérateurs 4h-12h × 2 opérateurs 12h-20h,
- Configuration 8 : 3 opérateurs 4h-12h × 3 opérateurs 12h-20h.

Pour gérer au mieux les files d'attente devant les machines, nous avons déterminés pour chaque configuration le Cmax et avons retenue la meilleure heuristique en fonction de ce critère.

- pour la Configuration 4, l'heuristique SPT+Johnson est la meilleure,
- pour la Configuration 5 (variante 2), l'heuristique SPT est un bon choix,
- pour la Configuration 8, on pourra utiliser indifféremment les heuristiques SPT ou l'heuristique hybride SPT+Johnson.

Il est possible d'introduire pour une configuration donnée, une alternative au choix de l'heuristique que nous proposons. Cependant, nos expériences ont montré que les configurations que nous avons présentées étaient suffisantes dans le cas de la population étudiée. En effet, même si parfois une autre heuristique améliore le Cmax (LAWR marche bien dans certains cas), le gain n'est que de quelques minutes et n'est donc pas significatif.
Cette étape nous a permis d'étiqueter les exemples en leur affectant une classe relative aux couples (calendriers/ressources humaines, heuristiques) retenus.

- Classe 1 (C1) : Config. 4 / Heuristique SPT+Johnson,
- Classe 2 (C2) : Config. 5 (variante 2) + Heuristique SPT,
- Classe 3 (C3) : Config. 8 + Heuristique (SPT ou SPT+Johnson).

Ces catégories sont donc les classes que le système d'apprentissage doit apprendre à reconnaître.
Comme nous l'avons vu précédemment, nous avons acquis de la connaissance sur l'atelier de production étudié, et obtenons ainsi une première base de données techniques composée d'exemples déterminés par des descripteurs numériques représentant la charge des machines associées à une classe de problèmes.
$y_1$ : la charge globale, $y_2$ le nombre d'O.F., $y_3$ à $y_6$ les charges respectives des machines de M1 à M6, et C la variable qualitative prenant les valeurs associées aux classes C1 C2 et C3 définies ci-dessus.
Nous avons discrétisé l'intervalle de valeurs de chaque variable, en 3 valeurs symboliques qui permet ainsi de généraliser les résultats. Valeurs désignant respectivement une estimation faible, moyenne ou forte de la valeur de la variable.
La Figure 4 montre un échantillon de l'ensemble de données ainsi obtenu en prenant pour exemple la semaine 1.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Semaine 1** | 146,43 | 16 | 24,41 | 36,11 | 18,72 | 22,73 | 28,00 | 16,46 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Semaine 1** | a | b | a | a | a | a | a | a |

**Figure 4.** *Discrétisation des variables numériques*

Le processus d'apprentissage peut maintenant être initié et nous présentons les résultats obtenus avec les arbres de décision.

### 4.4. *fouille de données et filtrage de la BDT*

Dans le système que nous avons mis en place, l'apprentissage porte d'une part sur les heuristiques gérant les files d'attente et d'autre part sur les aspects "Horaires/Ressources humaines" considérés.
L'étape suivante consiste à soumettre l'ensemble de données complet, d'une part, et l'ensemble de données réduit, d'autre part, à l'algorithme d'induction d'arbres de décision C4.5.

**Structure d'un arbre de décision**

Il existe trois éléments constituant un arbres de décision : les nœuds, les branches et les feuilles. Chaque nœud représente un sous-ensemble de la population d'apprentissage et est associé à une variable (attribut). De même chaque branche est associée à l'une des modalités de cette variable. Enfin, les feuilles, ou nœuds terminaux, représentent les différentes classes d'appartenance des objets.

La Figure 5 présente l'arbre obtenu avec C4.5. Notons que nous sommes intervenus sur l'algorithme afin de désactiver le fenêtrage et l'élagage qui sont actifs par défaut. Remarquons que cet arbre assez simple (bonne compréhensibilité) classe correctement reconnue à 88% de la population test (30 semaines de production) ce qui lui confère une assez bonne capacité de généralisation

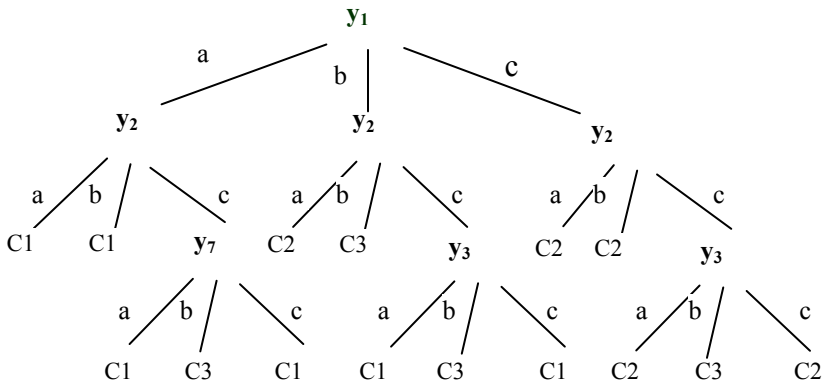


**Figure 5.** *Arbre de décision généré par C4.5 à partir de l'ensemble d'objets*

Nous constatons que les variables pertinentes de l'atelier sont $y_1$ la charge totale du flow shop hybride $y_2$ le nombre d'O.F., ainsi que $y_3$ et $y_7$.la charge respective des machines M1 et M5
Voici un exemple de règles de décision obtenues à partir de l'arbre
**Si** [y1=a] et [y2=a] **Alors** appliquer C1
Si la charge globale est faible et que le nombre d'O.F. est faible on applique la Config. 4 Heuristique SPT+Johnson
**Si** [y1=a] et [y2=c] et [y7=b] **Alors** appliquer C2
Si la charge globale est faible et que le nombre d'O.F. est élevé et que la charge de la machine 5 est moyenne alors on applique : Config. 5 (variante 2) + Heuristique SPT
**Si** [y1=c] et [y2=c] et [y3=b] **Alors** appliquer C3
Si la charge globale est élevée et que le nombre d'O.F. est élevé et que la charge de la machine 1 est moyenne alors on applique Config. 8 + Heuristique (SPT ou SPT+Johnson).
De plus, presque tous les cas mal classés sont affectés à une configuration permettant de finir à temps mais entraînant un coût financier plus important.

## 5. Conclusion

Nous exposons dans notre article une méthode permettant de prédire, avec une bonne précision, les divers paramètres d'un ordonnancement pour le pilotage d'atelier. La démarche proposée pour paramétrer un logiciel du marché, est basé sur un système d'apprentissage inductif associé à une étape de simulation et de résolution de problèmes. Elle a permis de créer de la connaissance sur l'atelier à

partir d'un minimum d'information. Cette démarche aboutit à l'obtention de règles d'ordonnancement dépendant de la configuration de l'atelier.
En intégrant au sein d'un logiciel de simulation d'ordonnancement, les outils d'extraction des connaissances à partir des données, c'est-à-dire un programme sur les arbres de décision, ainsi que l'algorithme de filtrage de variables et de données, une aide à la décision sous forme de règles expertes peut être proposée. De plus, une mise à jour régulière de la base de données techniques et l'application de méthodes d'apprentissage et de filtrage, permet de maintenir à jour le système d'aide à la décision.
Nous orientons actuellement nos recherches vers la prise en compte d'autres facteurs liés aux ressources humaines dans le cadre de l'ordonnancement, et mettons au point un algorithme unifiant les processus de filtrage de variables et d'exemples. Une fois mis en place, ces outils pourront être d'une grande utilité aux communautés ordonnancement et ECD.
Toutefois les outils proposés, les arbres de décision ne sont pas dynamiques et nécessitent de refaire tout le traitement quand de nouvelles données sont à simuler, en l'occurrence quand il y a un changement conséquent sur le nombre de ressources humaines dans l'atelier, un nombre de machines évoluant. Nous désirons rendre notre système d'aide plus interactif et le faire évoluer vers un système de raisonnement à partir de cas, qui permettrait de faire évoluer les données, de tenir compte de la connaissance du domaine et d'intégrer le décideur un peu plus dans la boucle de décision.

## 6. Bibliographie

Blasewicz J., Ecker K, Schmidt G., Weglarz J., Scheduling in Computer and Manufacturing Systems, Springer Verlag, Berlin heidelberg, 1993

Breiman, L.,Friedman, J. H., Olshen, R. A., and Stone, C. J. Classification and Regression Trees. Belmont, CA: Wadsworth, 1984.

Chebel-Morello B., Lereno E., Baptiste P. "A New Algorithm To Select Learning Examples from Learning Data", Proc. Of Intelligent Data Engineering and Automated Learning, LNCS/LNAI Series, Springer Verlag, 2000.

Chebel-Morello B., Michaut D.,Baptiste P.,"A knowledge discovery process for a flexible manufacturing system" Proc of the Emerging Technologies anf Factoty Automation, volume1 october 15 – 18 , 2001 Antibes, pp 652 - 659

Diday E., Brito P. "Introduction to Symbolic Data Analysis" Conceptual and Numerical Analysis of Data, (pp. 45-84) Ed. O. Opitz, Springer, Berlin Heidelberg New York, 1989.

Fayyad, U. M., Piatetsky-Shapiro, G., and Smyth, P. "From Data Mining to Knowledge Discovery: An Overview. Advances in Knowledge Discovery and Data Mining (pp. 1-34), 1996.

Geneste L., Domenech J.F., "Evaluation of job Shop Scheduling Heuristics in Workshop Subject to Disturbances, Proc. Of CIMAT'96 Computer Integrated Manufacturing and automation technology, Grenoble, France 1996.

Geneste L., Grabot B., “Implicit versus explicit knowledge representation for job shop scheduling”, International Journal of expert systems, research and applications, vol.10 1997, n°1, p 37-52

Giard V. Gestion de Production, Paris, Edition Economica, 1988.

Harrath Y., Chebel-Morello B., Zerhouni N., “Using learning to find patterns in genetic algorithm solutions to solve job-shop scheduling “Proc. for the International Conference on Industrial Engineering and Production Management, IEPM'03, sur CD ROM, 10 pages, 26-28 mai 2003, Porto, Portugal.

Hunt, E.B., Marin, J., Stone, P.J. Experiments in induction, New York Academic Press, 1966.

Huyet A. L., Paris J.L., Configuration and analysis of a multiproduct kanban system using evolutionary optimization coupled to machine learning, CESA’2003, Lille, 2003.

Kassou I., Berrada I., Pécuchet J.P., Object oriented methodology with neighbourhood search techniques and hybrid methods for scheduling problems. Journal européen des systèmes automatisés, Hermes, vol. 32, n° 4, 1998, .pp. 415- 430.

Lereno E., Chebel-Morello B., Baptiste P., "Système d’aide au paramétrage d’un logiciel en ordonnancement” 3° conférence Francophone de Modélisation et de simulation MOSIM’01 p 363-369, Troyes 2001

Liu, H., Motoda, H. Feature Selection for Knowledge Discovery and Data Mining Kluwer Acad. Publishers, 1998.

Michalsky, R. S. and Larson, J. B. "Selection of Most Representative Training Examples and Incremental Generation of VL1 Hypotheses : the underlying methodology and the description programs ESEL and AQ11", Report No 867, Dept. of Computer Science, University of Illinois, Urbana, 1978.

Michaut D., Lereno E., Morello B. and Baptiste P. "A Learning Procedure For Scheduling A Flexible Manufacturing System With A Noisy Learning Set", Proc. Of 2nd Asia-Europe Congress on Mechatronics, Japan, p. 692-696, 1998.

Paris J.L., Pierreval H., A distributed evolutionary simulation optimization approach for the configuration of multiproduct kanban systems. Computer Integrated Manufacturing, Vol 14,n°5, pp 421-430, 2001

Quinlan, J. R. C4.5: Programs for Machine Learning. San Francisco: Morgan Kaufmann, 1983.

Quinlan, J. R. "Learning Efficient Classification Procedures and Their Application to Chess End Games" In R. S. Michalsky, J. G. Carbonnel, and T. M. Mitchell (Eds.), Machine Learning: An Artificial Intelligence Approach. San Francisco: Morgan Kaufmann, 1993.

Talbi E. D., Geneste L., Grabot R., Meta-heuristics for optimal set-up of an industrial scheduling software, CESA’2003, 9-11 juillet 2003, Lille, France.

Talbi E. D., Geneste L., Grabot R, “algorithmes évolutifs pour le paramétrage d’un logiciel d’ordonnancement » 5° conférence Francophone de Modélisation et de simulation 1-3 sept 2004 Nantes, France p 67-74