

FCPPG : Générateur de planificateurs de tâches pour une cellule flexible de production.

Samia Aoudia ², Djamila Ouelhadj ¹ et Abdelgnani Souilah ¹

¹ Centre de Développement Des Technologies Avancées

128, Chemin Mohamed Gacem, El Madania, Alger, Algérie

² Institut National d'Informatique

B.P. 68M Oued smar, 16270 Alger, Algérie

Mots clefs

Cellule flexible de production, Planification de tâches, Générateur de planificateurs, Intelligence artificielle, Logique des prédicats, Programmation orienté objet.

Résumé

Dans cet article, nous décrivons un générateur de planificateurs de tâches dédié au cellules flexibles de production où des tâches d'usinage, d'assemblage et de manutention sont exécutées. Notre système s'inspire de l'approche systèmes experts. Cela nécessite la définition de trois modules superposés: un langage de description de la connaissance du domaine, un langage de description de l'application et un mécanisme de raisonnement. Le langage de représentation de la connaissance est une hybridation entre trois formalismes: les objets, la logique des prédicats et les règles de production. Le moteur d'inférence du premier ordre fonctionne en chaînage arrière, son mécanisme de filtrage est optimisé par des heuristiques.

Abstract

This paper describes a task planners generator dedicated to flexible production cells where tasks like machining, assembling and material handling are processed. Our system is inspired from expert system approach. This needs the definition of tree superposed moduls: a knowledge description language, a data or appliction description language and a reasoning mechanism. The knowledge description language is a hybridization between three formalisms: objects, predicat logic and production rules. The first order inference engine runs with backward chaining and incorporates a matching mechanism optimised with some developed heuristics.

1. Introduction

La recherche en productique tend vers une intégration de toute la chaîne allant de la conception jusqu'à la fabrication du produit fini et même sa commercialisation dans des systèmes informatiques. Ceci rentre dans le cadre d'une discipline communément appelée «Computer Integrated Manufacturing» (CIM) [1]. Parmi les maillons de cette chaîne nous retrouvons les systèmes générateurs de plans de tâches de fabrication. Nous nous intéressons, dans ce travail à

la planification de tâches dans les systèmes cellulaires de production et en particulier aux tâches effectuées à l'intérieur des cellules [9]. Ces tâches ont la particularité d'être complémentaires, d'avoir la pièce comme dénominateur commun et d'être liés à travers leurs ressources. Ces types de tâches sont, l'usinage, l'assemblage, manutention, contrôle de qualité, stockage, etc... .

La planification de ces types de tâches se faisait traditionnellement entièrement par des techniciens dans des bureaux de méthodes d'une manière séparée et séquentielle. Le processus prenait un temps très important alors que la cadence des exigences du marché en nouveaux produits est devenue beaucoup plus rapide. Cela a incité les chercheurs dans ce domaine à réfléchir sur des systèmes d'aide à la décision pour augmenter l'efficacité du technicien des méthodes et augmenter la flexibilité du système tout entier (en améliorant les autres éléments du réseau du système de production bien entendu). Il s'agissait de générer un programme formé d'un ensemble de tests sur les connaissances disponibles permettant de générer le plan recherché. Avec l'accroissement et la complication des connaissances utilisées qui a entraîné un nombre incontrôlable de tests, cette approche a très rapidement saturé les mémoires des ordinateurs. Par conséquent, il a été indispensable de faire appel à des techniques de l'intelligence artificielle pour générer les plans d'une manière intelligente. Son principe de base repose sur la séparation entre les connaissances et le raisonnement. Cette séparation a donné naissance à des systèmes très sophistiqués (systèmes experts, systèmes à base de connaissance, ...) utilisant une grande quantité de connaissances représentées avec plusieurs formalismes, et un mode de raisonnement efficace indépendant de la connaissance.

Dans cet axe de vision, nous proposons de réaliser un générateur de plans prenant en charge les problèmes de la planification, notamment ceux rencontrés dans les systèmes de production, et plus particulièrement ceux d'une cellule flexible de production. Ce système est basé sur les techniques de l'Intelligence Artificielle, en particulier l'approche système expert. Cela dans le but de pouvoir planifier simultanément plusieurs types de tâche sur un même produit (Concurrent Engineering).

La suite de cet article sera organisée comme suit: la section 2 positionne le problème de la planification des tâches dans les systèmes de production et présente l'approche de résolution. Les sections 3, 4, 5 et 6 présentent le générateur FCPPG, et on terminera par une conclusion.

La suite de cet article sera organisée comme suit: la section 2 positionne le problème de la planification des tâches dans les systèmes de production et présente l'approche de résolution. Les sections 3, 4, 5 et 6 présentent le générateur FCPPG, et on terminera par une conclusion.

2. Position du problème

Dans un système flexible de production, la planification est destinée à organiser le travail et concerne les équipements, les hommes, les matières, les informations techniques et procédurales. Cela se fait d'une manière hiérarchique avec des niveaux de détail décroissant et sur des périodes de temps décroissantes (fig.1) [8].

Notre travail concerne le niveau opérationnel et particulièrement la planification des tâches de production dans une cellule flexible. Une cellule flexible étant un ensemble de ressources à valeur ajoutée (machine d'usinage ou d'assemblage) et des ressources sans valeur ajoutée (éléments de manutentions et de stockage, éléments de contrôle d'exécution, élément de contrôle de qualité, ...) dédié à la fabrication d'une famille de types de produits. Cette famille regroupe des types de produits présentant des caractéristiques physiques similaires et nécessitant des ressources identiques lors du déroulement de leur processus de fabrication. La cellule est dite flexible car, d'une part, plusieurs types de produits sont produits dans la cellule à des cadences variables en

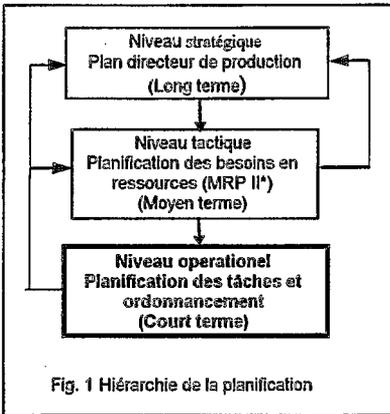


Fig. 1 Hiérarchie de la planification

fonction de la demande, et d'autre part, ces types de produits sont réalisés simultanément. Du fait que la cellule est flexible, ces trois principales tâches sont planifiées et replanifiées globalement ou partiellement si l'un des cas suivants se présente:

- ajout ou suppression d'un produit dans la famille,
- forte fluctuation de la demande de produit,
- ajout ou suppression d'une ressource dans la cellule,
- surgissement d'aléas dans la cellule,
- modification de l'agencement de la cellule.

3. Approche de résolution

Pour assurer une bonne coordination entre les différents types de plans à générer, et pour optimiser le temps de traitement et améliorer la qualité de la solution finale, nous nous proposons de concevoir un générateur de planificateurs de tâches FCPPG (Flexible Cell Process Planner Generator) prenant en charge les trois principaux types de tâches : usinage, assemblage, et maintenance. Ce système repose sur une approche IA, en particulier la méthodologie système expert. Cela suppose le développement d'un langage de description des connaissances du monde et d'un moteur d'inférences [3][4]. Le langage va permettre de décrire les connaissances du domaine de la production nécessaires à la planification, à savoir : les objets, les relations entre objets et les opérateurs de planification. Le moteur d'inférences est le raisonnement qui permet de générer le plan de tâches recherché. Notre système se veut ouvert et non spécifique, dans le sens

ou il comportera trois niveaux (fig. 2) à savoir le niveau système, le niveau expert et le niveau utilisateur.

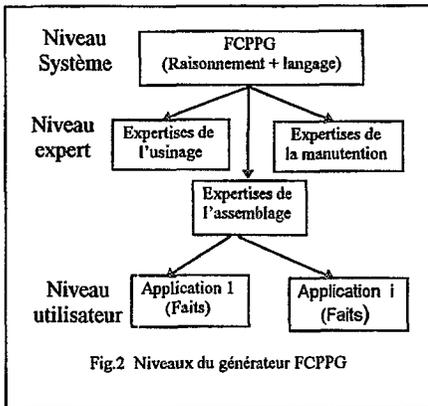


Fig.2 Niveaux du générateur FCPPG

Etant donnée une cellule flexible de production réalisant des opérations d'usinage et d'assemblage, dédiée à une famille donnée de types de produits, les données permettant la description du monde de la cellule sont les suivantes:

- D1- les dessins d'ensemble du produit assemblé (appelé dans la suite ensemble),
- D2- les dessins de définition des différentes pièces des différents ensembles à usiner dans la cellule,
- D3- la description des ressources à valeur ajoutée (machines d'usinage et d'assemblage),

D4- la description de ressources sans valeur ajoutée (éléments de maintenance et de stockage). Ces données seront utilisées, pour chaque type de tâche à planifier, comme suit :

D1+D2+D3 pour l'usinage, D1+D2+D3 pour l'assemblage et D1+D2+D3+D4 pour la maintenance.

4. Architecture générale du générateur FCPPG

L'architecture de notre générateur est schématiquement dans la figure 3.

5. Langage d'expression des connaissances.

La complexité et la variété des connaissances propres aux domaines de production (usinage, assemblage, manutention), nécessite la conception d'un langage adoptant une représentation hybride :

- représentation orientée objets pour la description des objets du monde,
- logique des prédicats pour l'expression des relations,
- règles de production pour la représentation des connaissances opératoires. Ces règles sont, à leur tour, exprimées par des variables (objets structurés) et des prédicats.

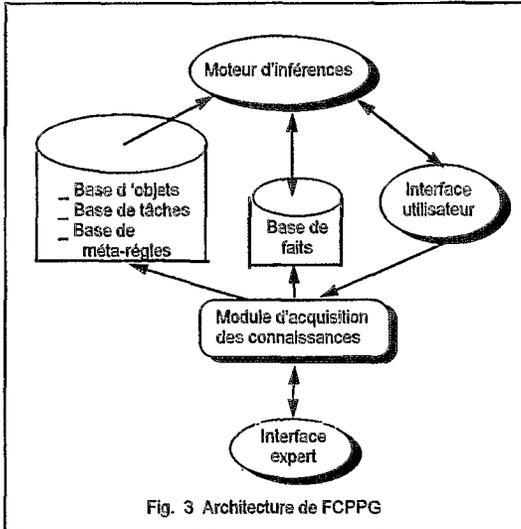


Fig. 3 Architecture de FCPPG

5.1. description des objets et des relations du monde: Base d'objets

Un objet du monde est représenté par un frame. Un frame est l'entité génératrice (prototype) capable d'engendrer les instances du frame par application d'un processus d'instantiation [2]. Un

```

((Nom frame )
 (type frame)
 (attribut1(type valeur))
 .....
 (attributn(type valeur))
 (liste-sous-frames))
    
```

exemple du prototype de frame est le suivant.

Le frame décrit un objet hiérarchiquement. Cette notion de hiérarchie est essentiellement utilisée lorsqu'on manipule des objets pouvant être subdivisés en un certain nombre de composants. La hiérarchie de composition (lien d'héritage) est schématisée par un arbre (fig. 4) . Un exemple de description du frame pièce est donné dans figure 5.

Une relation du monde est décrite par un prédicat suivi de la liste de ses arguments.

Exemple : SUR (pièce 1, pièce 2)

5.2. Description des connaissances opératoires

Les connaissances représentent le savoir-faire sur le domaine. On distingue deux types de connaissances opératoires :

- les tâches de planification (opérateurs),
- les méta_règles : règles de sélection de tâches.

Elles sont exprimées par des règles de production . Le bloc condition traduit la situation que doit vérifier le monde(état initial ou final) pour le déclenchement de la conclusion. Le bloc condition est formé d'un ensemble de prémisses. Deux prémisses successives sont implicitement liées par le connecteur ET. La partie condition sera vérifiée si toutes les prémisses qui le composent sont vérifiées, elle est fautive si l'une au moins de ses prémisses est fautive.

Une prémisses est soit : (i) un prédicat suivi de la liste de ses arguments, (ii) une contrainte qui est un test de comparaison de la forme : <état> <Frame>.att <comparateur> <valeur>.

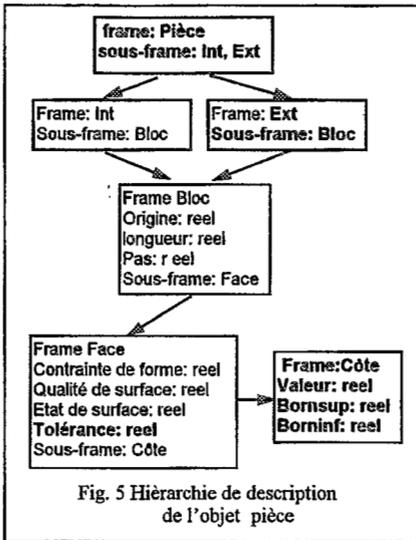


Fig. 5 Hiérarchie de description de l'objet pièce

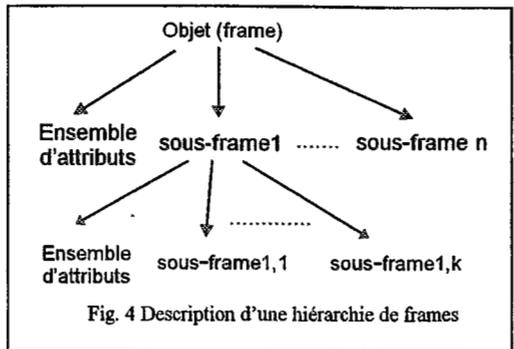


Fig. 4 Description d'une hiérarchie de frames

5.2.1. Description des tâches (opérateurs) de planification: base de tâches

Ces tâches représentent les opérateurs de planification. La solution de problèmes d'ordre pratique, notamment ceux du monde de la

production, nécessite l'utilisation d'une grande quantité de connaissances complexes, ce qui rend leur représentation et leur utilisation pratiquement impossible d'un seul coup, d'où la nécessité d'une description hiérarchique des opérateurs [7]. Notre modèle d'opérateur est hiérarchique dans le sens où une tâche peut contenir d'autres tâches, appelées sous-tâches qui ont rigoureusement la même syntaxe. La figure 6 montre une telle hiérarchie de tâches :

Les tâches les plus basses dans la hiérarchie, composées donc aucune d'autre sous-tâche, seront appelées tâches élémentaires. Le modèle générique d'une tâche est décrit ainsi:

```

  ((Nom Tache : arg1, ..., argn)
  (nom tache-mère)
  (priorité tache)
  (Si Préconditions Alors effets))
  
```

Les pré-conditions sont les conditions d'application de la tâche. La partie effets contient tous les effets entraînés par le déclenchement de la tâche sur l'environnement du problème.

Un changement peut être un ajout, une suppression de prédicat ou un changement de la valeur d'un attribut d'objet. Ce dernier changement est appelé affectation de valeur. L'exemple suivant (Tableau 1) illustre une tâche d'usinage: Un chambrage (intérieur) n'est effectué que si un premier perçage est déjà réalisé et si l'outil peut accéder à la surface à chambrer. Ses effets correspondent aux changements effectués sur la pièce: diminution des diamètres de toutes les faces touchées par l'opération.

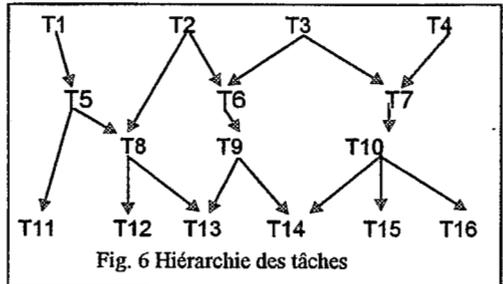


Fig. 6 Hiérarchie des tâches

Tâche : *chambrage*

Priorité 2

SI [(initial piece.exi.bloc-h.face-j.côte-k > 0) et (inf (initial piece.int.bloc-l.face-j.*côte.val) > 12)
et (initial piece.int.bloc-i.face-j.côte-k.val < final piece.int.bloc-i.face-j.côte-k)]

ALORS [(Initial piece.int.bloc-i.face-j.côte-k.val= finalpiece.int.bloc-i.face-j.côte-k.val)
et(Initial piece.int.bloc-i.face-j.côte-k+1.val= final piece.int.bloc-i.face-j.côte-k+1.val)
et(Initial piece.int.bloc-i.face-j+1.côte-k.val= final piece.int.bloc-i+1.face-j.côte-k.val)
et (Initial piece.int.bloc-i-1.face-j+1.côte-k.val= final piece.int.bloc-i.face-j.côte-k.val)]

où: h: numéro du dernier bloc ; i : indice du bloc concerné par le chambrage ; j : relatif au numéro de la face latérale j du bloc ; k : premier diamètre de la face j du bloc l : i, i+1, ..., n ; n étant l'indice du dernier bloc.

Tab. 1 Règle de sélection de la tâche de chambrage

La tâche "emmanchement", de l'assemblage (Tab. 2), réalisant une liaison "pivot glissant" sera déclenchée si : (la liaison n'est pas encore réalisée) et (l'ensemble contient déjà une liaison réalisée, l'une des deux pièces concernée par cette liaison, doit être dans l'ensemble) et (la liaison en question empêche une autre liaison d'être réalisée, cette dernière doit être déjà réalisée). Ses effets se traduisent par : (une augmentation du nombre de pièces à insérer dans l'ensemble) et (une mise à jour l'état de la pièce) et (la liaison une fois réalisée, elle sera empêchée par toute autre liaison).

Tâche: *emmanchement*

Priorité=1

SI [(initial ensemble.liaison_i.etat=0) et (sup (initial ensemble.*liaison.etat=0)
et(inf(initial ensemble.liaison_i.lieu_j.(pièce).ordre)=0)
et (final *empêche (ensemble.liaison_i, ?x / initial x.etat =1)]

ALORS
[(ensemble.liaison_i.etat =1) et (ensemble.ordre=initial ensemble.ordre+1)
et(ensemble.liaison_i.lieu_j.(pièce).ordre=initial ensemble.ordre)
et (supprim *empêche (?y, initial ensemble.liaison_i)],
TEL QUE [(i=1)]

Tab. 2 Règle de sélection de la tâche d'emmanchement

5.2.2. Description des règles de sélection de tâches: base de méta_règles

Pour faire une restriction sur l'ensemble des tâches qui interviennent dans la génération du plan recherché, on utilise des méta-règles, qu'on appelle règle de sélection de tâches. Le modèle générique d'une règle de sélection est décrit par les champs suivants :

où «condition» représente les conditions de sélection de la tâche. On retrouve dans la partie conclusion d'une règle les noms des tâches sélectionnées, chacune suivie de la liste de ses paramètres. Une sous-règle est l'ensemble des règles de sélection des sous-tâches de la tâche.

((Nom méta-règle) (condition) (conclusion) (sous-règles))
--

L'exemple du tableau 3 montre la règle de sélection de la tâche de chariotage cylindrique et ses sous-tâches.

5.3 Description des connaissances sur le problème: Base de faits

Elles sont relatives à l'énoncé du problème : description de tout l'environnement du problème avant l'application du processus de planification (état initial), et après application de celui-ci (état final). Ces deux états sont des instances des frames et des prédicats de la base d'objets.

Exemple: une instantiation du prédicat COUVRE(?x, ?y) est COUVRE(pièce1, pièce2), COUVRE(pièce2, pièce3).

Un exemple de description de l'état initial et de l'état final d'un ensemble de pièces à assembler est décrit dans les figures 7 et 8.

Règle 1: chariotage cylindrique1

SI [(finalpièce.ext.bloc_i.face_j.côte_k.val= final pièce.ext.bloc_i.face_j.côte_k+1.val)
 et (finalpièce.ext.bloc_i-1.face_j.côte_k+1.val< final pièce.ext.bloc_i.face_j.côte_k.val)
 et (finalpièce.ext.bloc_i+1.face_j.côte_k.val< final pièce.ext.bloc_i.face_j.côte_k.val)]

ALORS

Chariotagecylindrique(final pièce.ext.bloc_i.face_j)

Sous_règles : sous-règle1, sous-règle2

Sous-règle1

SI [(finalpièce.ext.bloc_i-1.face_j.etat-surface<6.3)
 et (final pièce.ext.bloc_i-1.face_j.qualité-surface μ 8)
 et (final pièce.ext.bloc_i-1.face_j.qualité-surface à 9)]

ALORS

[ebauche-charitagecylindrique(initialpièce.ext.bloc_i-1.face_j)
 finition-chariotagecylindrique(finalpièce.ext.bloc_i-1.face_j)]

Sous-règle2

SI [(final pièce.ext.bloc_i-1.face_j.etat-surface<6.3)
 et (final pièce.ext.bloc_i-1.face_j.qualité-surface μ 6)
 et (final pièce.ext.bloc_i-1.face_j.qualité-surface à 7)]

ALORS

[ebauche-chariotagecylindrique1(initialpièce.ext.bloc_i-1.face_j)
 demifinition-chariotagecylindrique1(initial pièce.ext.bloc_i.face_j)
 finition2-chariotagecylindrique1(finalpièce.ext.bloc_i-1.face_j)],

TEL QUE [(i > 1) et (i@) et (j=2) et (k=1)]

Tab. 3 Règle de selection de la tâche et ses sous-tâches

6. Moteur d'inférences de FCPPG

Le moteur d'inférences permet de déterminer la séquence de tâches qui sera appliquées sur l'état initial du monde pour atteindre l'état final en exploitant les connaissances de la base de faits et celles de la base de connaissances (opérateurs et méta-règles). L'originalité de ce moteur réside dans sa puissance de manipulation d'une structure hybride (objets, règles de production, logique des prédicats). Ce moteur se caractérise par :

1. un espace de recherche représenté en espace d'états, où chaque état correspond à une situation susceptible de se produire durant le processus de planification.
2. un raisonnement en chaînage avant,
3. une stratégie de contrôle en profondeur d'abord,
4. représentation des faits adoptant l'hypothèse du monde ouvert, qui stipule que tout fait doit être explicitement représenté dans l'ensemble de description des deux états d'un processus de planification: l'état initial et l'état final.

Tout fait peut être vrais, faux ou inconnu. Le raisonnement proprement dit du système, se réduit aux deux phases de restriction et de planification [6] (figure 9).

6.1. Restriction de l'ensemble de tâches

Cette phase permet de réduire l'ensemble des tâches à filtrer avec les faits de la base de faits dans l'étape de planification en utilisant les méta-règles (règles de sélection de tâches).

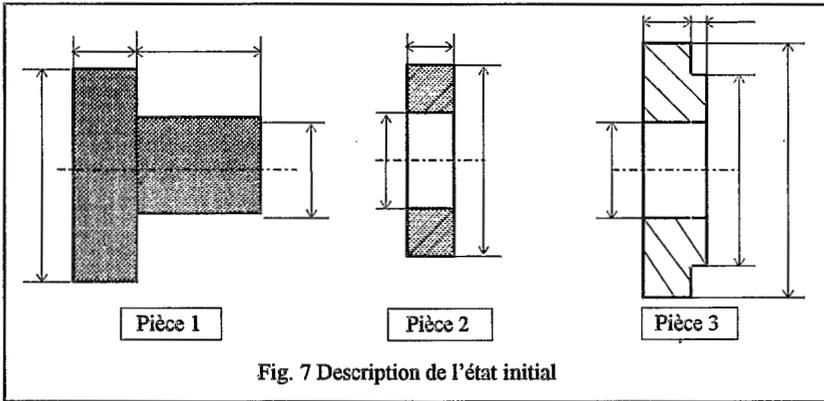


Fig. 7 Description de l'état initial

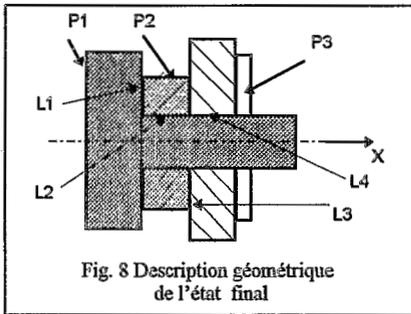


Fig. 8 Description géométrique de l'état final

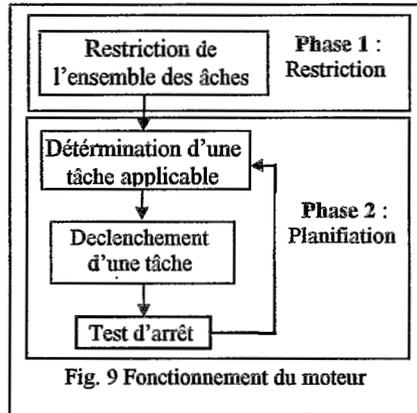


Fig. 9 Fonctionnement du moteur

6.2. Phase Planification

A ce niveau, la notion de hiérarchie entre les tâches est négligée en remettant tout à plat. Toutes les tâches et sous-tâches sont prises au même niveau.

a. Détermination d'une tâche déclenchable

Une tâche sera déclenchée si elle est sélectionnée par une méta-règle (la priorité de la tâche correspond à la priorité courante du système) ou si le filtrage de la partie condition de la tâche avec les faits de la base de faits est un succès.

b. Exécution (déclenchement) d'une tâche

Cette phase consiste à déclencher la tâche choisie lors de la phase précédente. Le déclenchement de la tâche consiste en la prise en compte de ses effets sur l'état courant du problème. Ces effets seront réellement pris en compte si la tâche ne possède aucune sous-tâche (cas d'une tâche élémentaire), ou si ses sous-tâches sont toutes déclenchées. Un effet est soit un ajout de prédicat, une suppression de prédicat, ou un changement de valeur de l'attribut d'un objet.

c. Test de condition d'arrêt

Un arrêt du raisonnement se voit nécessaire, si le plan recherché est atteint par l'atteinte de l'état final du monde. L'état final est atteint si les valeurs des attributs de tous les objets de l'état

courant sont égales à celles de l'état final, et l'ensemble des prédicats de l'état courant est identique à celui de l'état final.

L'arrêt du système n'est pas toujours déclenché après génération du plan de tâches, ce dernier peut être causé en cas d'un plan non réalisable. Un plan est dit non réalisable si aucune tâche ne peut être déclenchée avant d'atteindre la condition d'arrêt.

7. Conclusion

Le système FCPPG est destiné à la génération de planificateurs des tâches d'usinage, d'assemblage et de manutention dans une cellule flexible de production. Ce système se basant sur une approche I.A, dispose d'un langage de description des connaissances hybride assez puissant (objets, logique des prédicats, règles de production), et d'un moteur d'inférences d'ordre 1. Les interfaces utilisateur/expert sont développés sur l'environnement Windows offrant la possibilité d'utilisation de la souris et du multi-fenêtrage. FCPPG a été validé dans les domaines de l'usinage et de l'assemblage satisfaction.

Annexe

Session de génération d'un plan d'usinage d'une pièce de révolution.

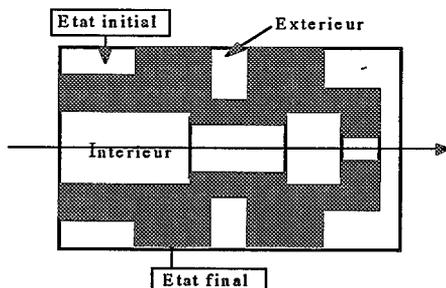


Fig.10 Description de l'état initial et final d'une pièce

Résultats de la phase de restriction

1. chariotage-cylindrique(*final piece.ext.bloc2.face2, final piece.ext.bloc4.face2, final piece.ext.bloc2.face2*)
2. dressage de face(*final piece.ext.bloc5.face3*)
3. draissage-angle2 (*final piece.ext.bloc5.face2, final piece.ext.bloc4.face3*)
4. rainurage (*final piece.ext.bloc2.face3, final piece.ext.bloc3.face2, final piece.ext.bloc4.face1, final piece.ext.bloc1.face2, final piece.ext.bloc2.face1*)
5. centrage (*final piece.ext.bloc5.face2*)
6. filetage (*final piece.ext.bloc5.face2*)
7. tronçonnage (*final piece.ext.bloc1.face1*)
8. perçage(*final piece.int.bloc4.face2*)
9. alésage cylindrique (*final piece.int.bloc4.face2*)
10. chambrage (*final piece.int.bloc2.face2*)
11. gorgeage (*final piece.int.bloc3.face2, final piece.int.bloc3.face1, final piece.int.bloc1.face2, final piece.int.bloc1.face1, final piece.int.bloc1.face3*)

12. taraudage (final piece.int.bloc4.face2)

Résultat de la phase de planification

10 draissage-face (final piece.ext.bloc5.face3)

20. chariotage-cylindrique(final piece.ext.bloc2.face2)

30. rainurage(final piece.ext.bloc1.face2 final piece.ext.bloc2.face1, final piece.ext.bloc2.face3, final piece.ext.bloc3.face2 final piece.ext.bloc4.face1)

50. draissage-angle (final piece.ext.bloc5.face2, final piece.ext.bloc4.face3)

60. chariotage-cylindrique(final piece.ext.bloc4.face2)

70. filetage (final piece.ext.bloc5.face2)

80. centrage (final piece.ext.bloc5.face2)

90. perçage (final piece.int.bloc4.face2)

100. gorgeage (final piece.int.bloc1.face2, final piece.int.bloc1.face1, final piece.int.bloc1.face3, final piece.int.bloc3.face2, final piece.int.bloc3.face1)

120. chambrage (final piece.int.bloc2.face2)

130. alésage-cylindrique (final piece.int.bloc4.face2)

140. taraudage (final piece.int.bloc4.face2)

150. tronçonnage (final piece.ext.bloc1.face1)

Remerciements: Les auteurs tiennent à remercier Monsieur Mahfoud Guerbas pour l'aide précieuse qu'il leur a fournie. Ils tiennent également à remercier les examinateurs de ce travail pour leurs conseils enrichissants.

Bibliographie

[1] R. Bonetto, «Les ateliers flexibles de productions», Editions Hermes, 1987.

[2] D. Colnet, D. Léonard et K. Tomber, «Les langages à objets», Editions InterEdition, 1989.

[3] J. L. Laurière, «Intelligence Artificielle», Tome, Editions Eyrolles, 1987.

[4] A. Tate, J. Hindler, M. Drummond. «AI Planning systems and Techniques », ----, 1990.

[5] P. Padilla, A. Thely, «Guide de fabrication mécaniques», Edition Dunod, 1981.

[6] M. C. Rousset, «Tango: Moteur d'inférences pour une classe de systèmes avec variables», Thèse de doctorat, ----, 1983.

[7] H. Laruelle, «Planification temporelle et exécution de tâches en robotique», Thèse de doctorat, Université Paul Sabatier, 1994.

[8] V. Giard «Gestion de production», Edition Economica, 1987.

[9] J.-M. Proth et A. Souilah, «Agencement des systèmes cellulaires de fabrication», Rapport de recherche INRIA n° 1442, Juin 1991.