

## CHAPITRE IV

### REALISATION DU LOGICIEL

#### 1. Problématique :

En raison de l'importance économique considérable de problème d'ordonnancement, et la complexité de solution, l'automatisation de ce genre de problème est d'un grand intérêt. C'est pourquoi nous développons.

#### Mais c'est quoi Ordo ?

Ordo c'est un logiciel propose une interface homme-machine fondé sur une représentation simple permettant à l'utilisateur d'obtenir de manière prédictive un planing d'exécution d'un série des tâches.

L'objectif du logiciel est de trouver une solution approché de bonne qualité pour un problème d'ordonnancement à une machine avec le critère de la minimisation de la somme pondérée des retards (le critère minimiser  $\sum W_i T_i$  ).

L'utilisateur donne le nombre du tâches, les données initial sont allouées à l'utilisateur par l'édition de commande qui contient la liste de tâches est leurs caractéristique (durée d'exécution et date d'échue).

Le logiciel dispose également d'une zone de visualisation les tâches qui sont ordonnancées avec notre heuristique, plus de ça l'utilisateur peut visualiser la solution de problème avec la méthode recherche tabou.

En réalité Ordo offre 18 séquences : trios (03) séquences initiale (aléatoirement, avec le règle EDD, et avec le règle SWPT) \* six (06) méthode de voisinages (permutation simple de deux tâches adjacentes, permutation simple de deux tâches, voisinage par insertion de tâches, voisinage par insertion block de tâches, voisinage par hybridations 1 et 2 ). Il spécifie à chaque séquence leur somme pondérée des retards.

## 2. présentation de système et l'objectif de l'étude

**Objectif de l'étude :** Tout d'abord, nous pouvons rappeler quelques définitions :

- Ordonnancement un ensemble de tâches, c'est programmer leur exécution en leur allouant les ressources requises et en fixant les dates de début.
- Une ressource est un moyen technique ou humain, dans notre étude les ressources sont des machines.
- Une tâche (opération) est un travail élémentaire dont la réalisation nécessite un certain d'unités de temps (sa durée).

Notre but est de proposer un logiciel conduit à donner un séquence initiale d'un ensemble des tâche qui permet de minimiser la somme pondérée des retards, ce problème est très complexe, nous proposons un algorithme efficace pour résoudre des problèmes à une machine.

La solution (séquence des tâches) obtenu représentée sur l'interface de notre logiciel par un tableau.

### Hypothèse est contrainte de fonctionnement :

- La ressource est une machine disponible à l'instant initial ( $\Delta = 0$ ).
- Les tâches non préemptives (la machine n'exécute qu'une tâche à la fois).
- Les temps de préparation des tâches sont inclus dans leur durée.
- Le problème n'est pas répétitif.
- Les tâches toutes disponible à l'instant initial ( $r_i=0$ ).

## 3. conception du logiciel :

Nous retenons comme méthode de conception la méthode SADT (Structured Analysis and Design Technique). Nous utilisons les actigrammes SADT : les boîtes représentent les activités, les flèches modélisent les flux entre les activités, les mécanismes et les contrôles.

Dans cet actigramme on décompose le système en composants plus simples.

Où les boîtes représentent les procédures, les flèches indiquent les flux de données.

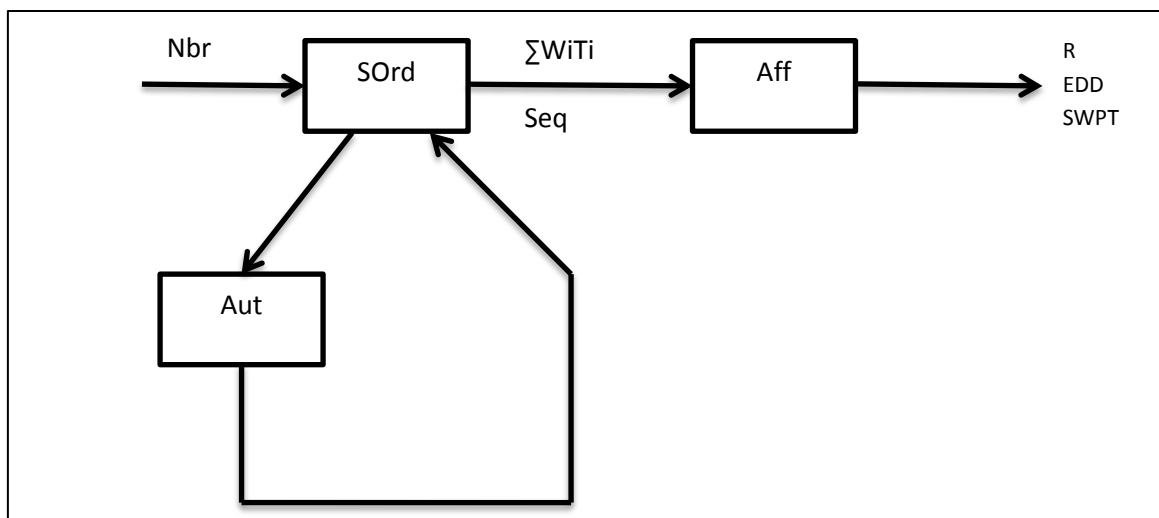


Figure 16 : actigramme de conception du logiciel

Dans la suite on donne les significations des codes :

code	signification	code	Signification
Nbr	Nombre des tâches	Aut	Chargement automatique(les caractéristiques des tâches)
SOrd	Le procédure qui résoudre le problème	R	Solutions avec séquence initial généré aléatoirement
Seq	La séquence des tâches	EDD	Solutions avec séquence initial généré par la règle EDD
Aff	Affichage des résultats	SWPT	Solutions avec séquence initial généré par la règle SWPT
$\sum WiTi$	La somme pondérée des retards		

#### 4. programmation :

##### Choix de langage :

Nous choisissons Delphi version 7 comme langage de programmation.

##### Qu'est-ce que Delphi ?

Delphi est un environnement de programmation visuel orienté objet pour le développement rapide d'application (RAD). Il permet de créer des applications Microsoft Windows très efficaces, avec un minimum de codage manuel.

Delphi fournit tous les outils qui sont nécessaires pour développer, tester, déboguer et déployer des applications, incluant une importante bibliothèque de composants réutilisables, un ensemble d'outils de conception, des modèles d'applications et de fiches, ainsi que des experts de programmation. Ces outils simplifient le prototypage et réduisent la durée du développement.

#### 5. Test du logiciel :

Après le développement de chaque procédure on fait des tests sur la validité aussi les liens avec les autres procédures. Dans le cas général on prend des exemples aléatoires, par contre dans les cas particuliers on fait des tests étudiés.

En fin nous prend comme test l'exemple décrit dans le chapitre 3, les résultats obtenus sont raisonnables.

## 6. Interface du logiciel :



Figure 17 : Interface du logiciel

## 7. Guide de l'utilisateur

A la première étape, cliquez sur le bouton *démarrer* pour afficher l'interface d'initialisation

Figure 18: initialisation les données

A la deuxième étape, vous avez le choix d'avoir le nombre souhaité des tâches et ainsi que celui des périodes et leurs durées, ensuite cliquez sur le bouton *remplir le tableau* pour générer les caractéristiques des tâches et le début et la fin de chaque période qui vont

s'afficher sur deux tableaux , l'un est conçu pour les caractéristiques des tâches et l'autre l'est pour le temps de l'indisponibilité .

Initialisation

Nombre de tâches (n) 10    Nombre de periodes (m) 5  
 Durée de service max 10    Durée d'une periode (l) 1

Remplir le Tableau

Tableau des données générées :

n	1	2	3	4	5	6	7	8
pi	1	3	4	5	1	3	8	9
wi	1	3	2	1	9	10	4	8
di	10	3	17	6	15	19	10	17

Tableau du temps d' indisponibilité :

si	6	12	18	24	30
ti	7	13	19	25	31

Nombre d'itérations : 1    Lancer le Logiciel

Traitement

Retour

Figure 19 : remplir les données

A la troisième étape, vous pouvez choisir le nombre souhaité d'itération, vous cliquez ensuite sur le bouton *lancer logiciel*

Initialisation

Nombre de tâches (n) 10    Nombre de periodes (m) 5  
 Durée de service max 10    Durée d'une periode (l) 1

Remplir le Tableau

Tableau des données générées :

n	1	2	3	4	5	6	7	8
pi	2	1	8	8	7	7	7	8
wi	2	6	7	6	6	10	3	5
di	22	30	24	23	6	5	28	24

Tableau du temps d' indisponibilité :

si	10	20	30	40	50
ti	11	21	31	41	51

Nombre d'itérations : 1000    Lancer le Logiciel

Traitement

Retour

Figure 20 : début traitement

Une fois la barre de progression est remplie, une fenêtre du résultat final s'affiche

The screenshot shows a window titled 'Resultat' with three main sections, each corresponding to a different initial sequence. Each section contains a table of results for various neighborhood techniques.

Section	Sequence Initiale	Technique	Solution Finale
séquence initiale aléatoire solution initiale 1494	6 1 5 3 2 4 8	voisinage par permutation de deux tâche adjacentes	863
		voisinage par permutation de deux tâche	863
		voisinage par insertion de tâches	863
		voisinage par insertion de block de tâches	863
		voisinage hybride1	863
		voisinage hybride2	863
séquence initiale générée à l'aide de la règle EDD [Earliest Due Date] solution initiale 1185	6 1 5 3 2 4 8	voisinage par permutation de deux tâche adjacentes	863
		voisinage par permutation de deux tâche	863
		voisinage par insertion de tâches	863
		voisinage par insertion de block de tâches	863
		voisinage hybride1	863
		voisinage hybride2	863
séquence initiale générée à l'aide de la règle SWPT (Shortest Weighted Processing Time) solution initiale 1950	6 1 5 3 2 4 8	voisinage par permutation de deux tâche adjacentes	863
		voisinage par permutation de deux tâche	863
		voisinage par insertion de tâches	863
		voisinage par insertion de block de tâches	863
		voisinage hybride1	863
		voisinage hybride2	863

Buttons: Print, Retour

Figure 21 : interface des résultats

Les résultats obtenus représentés sur les tableaux ci-dessus répartis sur trois parties qui la séquence initiale générée aléatoirement, la séquence initiale générée à l'aide de la règle EDD et la séquence initiale générée à l'aide de la règle SWPT, dont chacune comprend six séquences optimales et que chaque séquence optimale se résulte d'une technique de voisinage indépendante, ces techniques apparaissent sur la figure ci-dessus (figure 21).