OrdoNet, un outil de modélisation et d'analyse des graphes potentiel-tâche sous Matlab

Bruno Sareni[1], Gérard Fontan[1,2], Elodie Chanthery[2], Stéphane Caux[1]

{Gerard.Fontan, Bruno.Sareni, Stephane.Caux}@enseeiht.fr
[1]INPT-ENSEEIHT
Département Génie Electrique et Automatique
2 rue Camichel,
31 071 Toulouse Cedex, France

{Gerard.Fontan, elodie.chanthery}@laas.fr
[2]LAAS, CNRS
7 avenue du Colonel Roche,
31 077 Toulouse Cedex 4, France

RESUME : Dans cet article, nous présentons le logiciel OrdoNet, une interface développée sous Matlab pour la représentation et l'analyse des graphes potentiel-tâche. Cet outil a été spécifiquement conçu dans une optique pédagogique, pour aider les étudiants à mieux maîtriser la modélisation par graphes et les concepts liés à la planification ou à l'ordonnancement d'activités. OrdoNet est actuellement utilisé au département Génie Electrique et Automatique de l'ENSEEIHT dans le cadre d'une manipulation de travaux pratiques en Automatique, illustrant les enseignements théoriques liés aux graphes et à l'ordonnancement de tâches.

Mots clés : Graphes, Planification et ordonnancement, Algorithme de Ford

1 INTRODUCTION

La commande de systèmes complexes est généralement articulée en deux niveaux. Le niveau « inférieur » est celui des automatismes « élémentaires », qui commandent par exemple des actionneurs à partir de renseignements fournis par des capteurs. Le niveau « supérieur » traite de la commande des activités du système en considérant les différents éléments techniques (automatismes élémentaires, actionneurs, processus). La commande des activités (ou tâches) s'efforce de proposer des politiques de commandes aux automatismes élémentaires de manière à assurer un fonctionnement cohérent et optimisé de l'ensemble du système, ceci en tenant compte des contraintes techniques.

Un ingénieur en Automatique doit donc bien connaître ces deux niveaux de commande et maîtriser les notions associées à la planification et l'ordonnancement de tâches. Dans ce contexte, la maîtrise de la théorie des graphes et des algorithmes associés est indispensable. Pour faciliter la compréhension de cette partie « conceptuelle », parfois difficile à assimiler par les étudiants, nous avons introduit une manipulation en travaux pratiques, illustrant l'ordonnancement d'un réseau ferroviaire. Cette manipulation est basée sur l'utilisation du logiciel OrdoNet, interface développée sous Matlab au cours d'un « projet long »[1] par Thierry Peynot, Noëlle Bailbé et Elodie Chanthery, élèves ingénieurs de 3ème année en 2002. Cette interface permet la description, la représentation et l'analyse de graphes potentiel-tâche. Ce type de graphe [1-3] modélise un problème d'ordonnancement où chaque sommet x_i représente une tâche exécutée à la date t_i, et où un arc (x_i, x_j) de longueur a_{ij} représente une contrainte de réalisation entre les tâches i et j formulée sous forme d'une inégalité de potentiel ($t_j - t_i \geq a_{ij}$).

[1] Les « projets longs » ENSEEIHT sont des projets tutorés en 3ème année à caractère industriel, pédagogique ou de recherche, d'un volume horaire de 60 heures.

Il existe des outils du commerce gratuits (GanttProject par exemple [4]) offrant des fonctionnalités semblables à celles d'OrdoNet mais dont le code n'est pas forcément accessible ou documenté. C'est une des raisons qui nous a poussés à développer notre propre outil, sous Matlab, un langage assez bien maîtrisé par les étudiants.

Dans la deuxième partie de cet article, nous présentons plus précisément les caractéristiques et les potentialités du logiciel OrdoNet. Puis, nous décrivons dans la troisième partie la manipulation de TP développée autour de cet outil.

2 ORDONET

2.1 Présentation de l'application

Une fenêtre de contrôle (voir figure 1) permet d'accéder aux différentes fonctionnalités de l'application notamment :

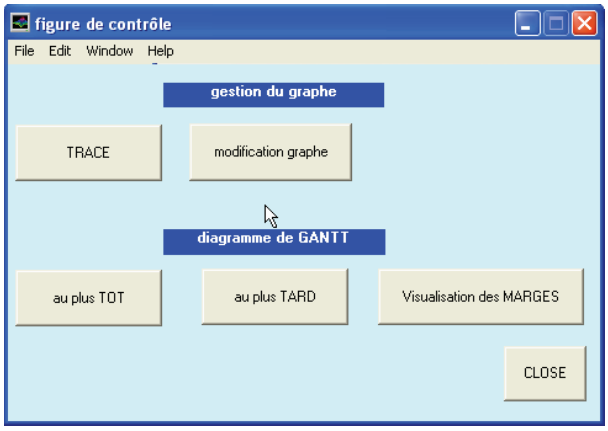

fig 1 : Fenêtre de contrôle général de l'application

- le calcul des dates d'exécution au plus tôt et des dates d'exécution au plus tard associées aux différentes tâches (ou sommets) du graphe.

- l'affichage du graphe, selon une structure définie par l'utilisateur ou une structure faisant apparaître les différents niveaux du graphe.
- l'affichage des diagrammes de Gantt selon un lancement des tâches au plus tôt ou au plus tard ou intégrant ces deux planifications de manière à faire apparaître les marges.

L'architecture logicielle de l'application est donnée à la figure 2. Les différentes fonctions apparaissant sont définies dans les paragraphes suivants.

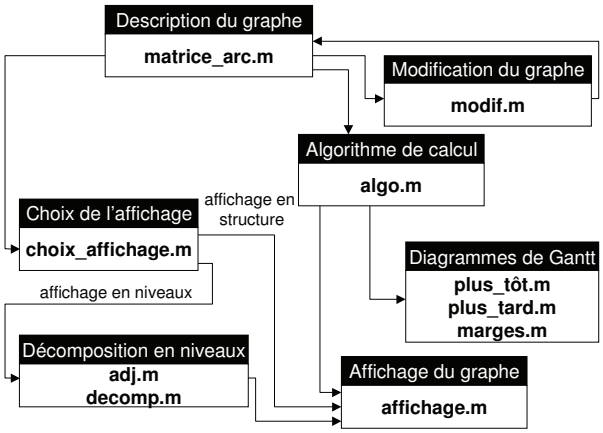

fig 2 : Architecture de l'application

2.2 Description du graphe

Le développement d'une interface graphique spécifique sous Matlab pour la saisie d'un graphe donné étant relativement lourd, nous avons préféré une description matricielle simple. Sous OrdoNet, un graphe est donc défini par une matrice symbolisant l'ensemble des arcs dans un fichier (**matrice_arc.m**), les tâches étant numérotées de 1 à *m* (*m* désignant le nombre total de tâches, incluant les taches « fictives » de début et de fin utilisées pour repérer la durée de réalisation de l'ensemble des tâches). Cette matrice d'arcs est composée de *n* colonnes (*n* désignant le nombre total d'arcs dans le graphe) et quatre lignes :

- la 1ère ligne contient pour chaque arc le numéro du sommet origine
- la 2ème ligne contient pour chaque arc le numéro du sommet extrémité
- la 3ème ligne contient la longueur de l'arc a_{ij} (durée de la tâche ou de la contrainte de séquencement entre deux tâches)
- la 4ème ligne contient une valeur binaire *u* précisant si l'arc est relatif à une durée de réalisation d'une tâche (*u*=0) ou à une contrainte de séquencement entre deux tâches (*u*=1). Cette variable est utilisée dans le logiciel pour faire la distinction entre les tâches (représentées par des barres horizontales) et les contraintes de séquencement entre tâches (représentées par un trait en pointillé, reliant les tâches en question)

A titre d'exemple, nous illustrons l'organisation des tâches d'un robot manipulateur dans une usine d'embouteillage d'eau. Le robot commence par amener une bouteille vide (durée : 6s) prise dans un stock au lieu de remplissage, puis il la remplit (durée : 20s). En parallèle avec ces actions, le robot (à plusieurs bras manipulateurs) va chercher un bouchon dans un stock (durée : 4s) et imprime aussi l'étiquette (durée : 10s). Une fois le bouchon à disposition et la bouteille pleine, le robot ferme celle-ci (durée : 5s). Le collage de l'étiquette (durée : 20s) débute lorsque la bouteille est à moitié pleine de façon à éviter un renversement. Le cahier des charges lié à cette organisation de tâches est résumé dans le tableau 1. Le graphe correspondant est représenté figure 3 et une matrice d'arc associée M_ARC, à implanter dans OrdoNet pour décrire le problème, est donnée ci-après :

Tâche	Durée	Contrainte
1 – Début	-	
2 – Prise bouteille	6 s	
3 – Remplissage	20 s	Tâche 2 achevée
4 – Prise bouchon	4 s	
5 – Fermeture	5 s	Tâches 3 et 4 achevées
6 – Impression	10 s	
7 – Collage	20 s	Tâche 6 achevée Tâche 3 avancée à moitié
8 – Fin	-	

Tableau 1 : Exemple de cahier des charges

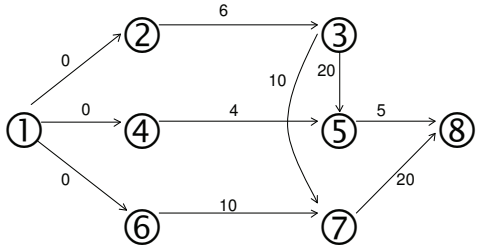

fig 3 : Graphe relatif au cahier des charges du tableau 1 (chaque sommet du graphe est associé à une tâche, repérée par un numéro dans la première colonne du tableau)

$$\text{M\_ARC} = \begin{pmatrix} 1 & 1 & 1 & 2 & 4 & 6 & 3 & 5 & 7 & 3 \\ 2 & 4 & 6 & 3 & 5 & 7 & 5 & 8 & 8 & 7 \\ 0 & 0 & 0 & 6 & 4 & 10 & 20 & 5 & 20 & 10 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

On remarquera qu'il est possible d'inverser des colonnes de la matrice sans changer le problème.

2.3 Calcul des dates d'exécution

Le calcul des dates d'exécution au plus tôt et des dates d'exécution au plus tard est réalisé à l'aide de l'algorithme de Ford [5]. Deux potentiels $V(x_i)$ et $W(x_i)$ sont associés à chaque sommet x_i du graphe. Ces potentiels représentent respectivement le plus long chemin dans le graphe du sommet « début » au sommet x_i et le plus long chemin du sommet x_i au sommet « fin ». Le potentiel $V(x_i)$ est calculé conformément au pseudo-code donné figure 4.

Le potentiel $W(x_i)$ est calculé de manière duale en initialisant tous les sommets à $-\infty$ et en considérant $W(x_{fin}) = 0$. Les dates d'exécution au plus tôt et au plus tard sont respectivement données par $V(x_i)$ et $L - W(x_i)$ où $L = V(x_{fin})$ désigne la longueur du chemin critique.

Etape 1 : Initialisation
Initialiser le potentiel de chaque sommet à
$V(x_i) = -\infty \ \forall\ x_i \neq x_{début}$ et $V(x_{début}) = 0$
Etape 2 : Amélioration des potentiels
Modification = 0
Pour tout arc (x_k, x_i) du graphe
Si $V(x_k)$ + longueur $(x_k, x_i) > V(x_i)$
$V(x_i) = V(x_k)$ + *longueur* (x_k, x_i)
Prédécesseur $(x_i) = x_k$
Modification = 1
Fin Si
Fin Pour
Revenir à l'étape 2 tant que *Modification* = 1

fig 4 : Pseudo-code de l'algorithme de Ford

2.4 Affichage du graphe

Dans OrdoNet il est possible d'afficher le graphe sous deux formes distinctes :

- soit sous une forme de structure prédéfinie par l'utilisateur. Il est alors possible de choisir la position des sommets sur une grille « virtuelle ». Pour cela, on définit dans la fenêtre de commande de Matlab une matrice de structure dont les éléments représentent les sommets du graphe. Des zéros sont ajoutés pour compléter la matrice et matérialiser les emplacements vides dans la grille. Dans le cas de l'exemple donné figure 3 la matrice de structure est donnée ci-après :

$$G_{structure} = \begin{pmatrix} 0 & 6 & 7 & 0 \\ 1 & 4 & 5 & 8 \\ 0 & 2 & 3 & 0 \end{pmatrix}$$

  On notera que cette représentation matricielle décrit les sommets du graphe « de bas en haut » et de « gauche à droite ». Ayant défini cette matrice, OrdoNet trace le graphe selon la structure donnée figure 3 (voir figure 5).

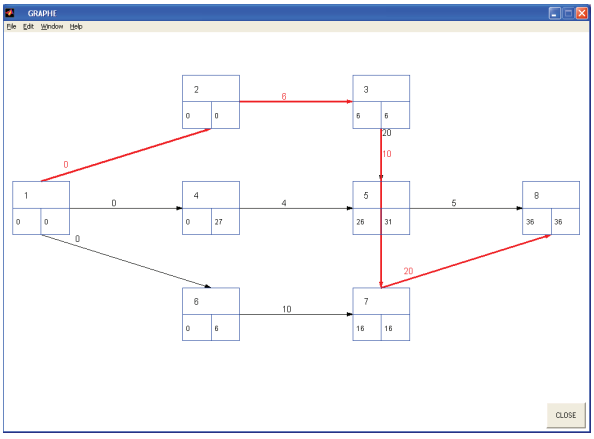

fig 5 : Affichage du graphe dans OrdoNet selon la structure de la figure 3

- soit sous une forme faisant apparaître une décomposition en niveaux du graphe, obtenue à partir de la matrice d'adjacence [1-2]. Cette représentation a l'avantage de montrer clairement le séquencement des tâches, dans certains cas, montrant les tâches pouvant être exécutées en parallèle (tâches associées à un même niveau). La figure 6 illustre le tracé du graphe sous OrdoNet pour l'exemple décrit précédemment.

Pour les deux représentations (en structure ou en niveaux), OrdoNet fait apparaître les dates d'exécutions relatives aux différentes tâches et le chemin critique, en marquant en rouge (et en trait fort) les arcs qui lui sont associés (tâches ou contraintes de séquencement critiques).

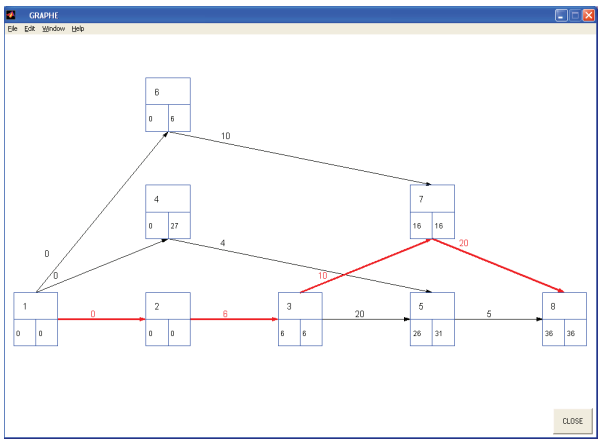

fig 6 : Affichage du graphe décomposé en niveaux

2.5 Tracé des diagrammes de Gantt

Une fois la matrice d'arcs définie et le calcul des dates d'exécution réalisé, OrdoNet permet de tracer les diagrammes de Gantt associés. A titre d'illustration, nous donnons le diagramme de Gantt faisant apparaître les marges relatives à chaque tâche pour l'exemple précédent (voir figure 7).

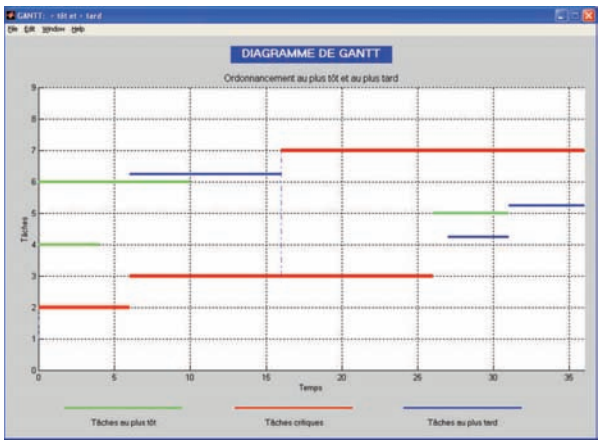

fig 7 : Affichage du diagramme de Gantt (visualisation de l'exécution au plus tôt et au plus tard)

On remarquera que les tâches sont symbolisées en trait fort (en vert pour une exécution au plus tôt, en bleu pour une exécution au plus tard et en rouge lorsque la tâche est critique). Les contraintes de séquencement entre les tâches figurent quant à elles en pointillées.

2.6 Modification du graphe initial

Enfin, OrdoNet offre la possibilité de faire évoluer le graphe défini par sa matrice d'arcs. Il est ainsi possible d'ajouter ou supprimer des arcs entre tâches existantes, voire de modifier la longueur d'un arc reliant deux tâches dans le graphe existant. Cette procédure se fait sans avoir à redéfinir la matrice d'arcs.

3 UNE MANIPULATION DE TP SUR LES GRAPHES ET L'ORDONNANCEMENT

3.1 Cahier des charges de la manipulation

L'objectif du TP est de modéliser et gérer le parcours de 3 trains circulant sur un micro-réseau ferroviaire.

Le réseau ferroviaire est découpé en tronçons (ou sections) élémentaires repérés par des numéros (voir figure 8). Nous précisons qu'il existe en salle de TP une maquette réelle de ce réseau, pilotée par automates programmables. Celle-ci est utilisée pour la mise en application des concepts liés aux automatismes logiques. Le choix d'une manipulation relative à la gestion du réseau s'est donc justifié pleinement, pour illustrer la planification des activités (couche de niveau hiérarchique supérieure aux automatismes « élémentaires »)

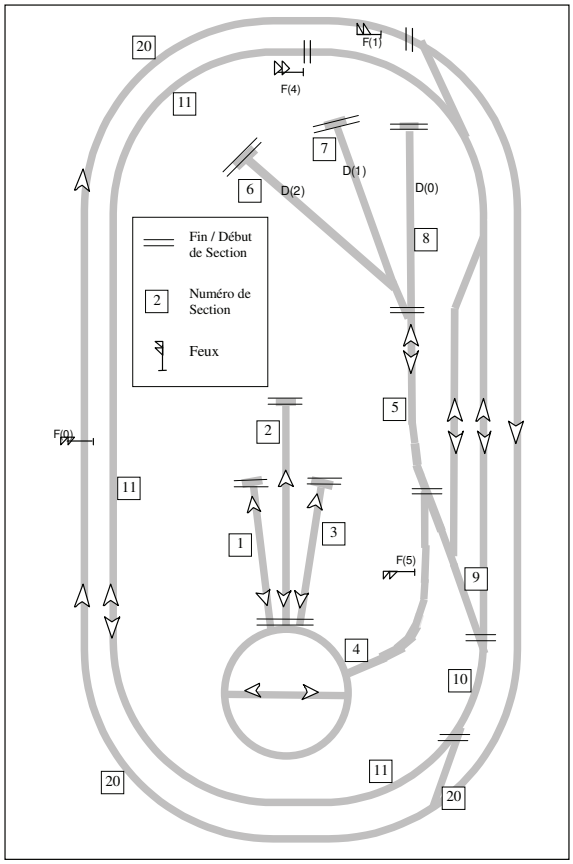

fig 8 : Micro-réseau ferroviaire

Les parcours des trains sont décrits par la succession des tronçons élémentaires traversés (par exemple 1, 4, 5, 6…). Outre l'ordre de parcours de ces tronçons pour un train donné, il est nécessaire d'assurer l'absence de conflit entre les trains. A tout instant, il ne peut y avoir qu'un seul train sur un tronçon donné. Les trois trains que l'on désignera par A, B et C doivent effectuer les parcours suivant :

- la locomotive du train A, stationnée dans la section 1, va chercher ses wagons dans la section 6 puis passe par le tronçon 9 pour aller effectuer le tour du « périphérique intérieur » et s'arrêter au feu F(4).
- la locomotive du train B, stationnée dans la section 2, va chercher ses wagons dans la section 7 puis passe par le tronçon 9 pour aller effectuer le tour du « périphérique extérieur » et s'arrêter au feu F(1)
- la locomotive du train C, stationnée dans la section 3, va chercher ses wagons dans la section 8 puis revient à son garage dans la section 3.

Les séquences des sections traversées par les trois trains sont résumées dans le tableau 2.

Train	Sections empruntées
A	1, 4, 5, 6, 5, 9, 10, 11
B	2, 4, 5, 7, 5, 9,10, 20
C	3, 4, 5, 8, 5, 4, 3

Tableau 2 : Parcours des trains

Les trains parcourent les différents tronçons à une vitesse différente, le train A étant le plus lent et le train B le plus rapide. Les durées de parcours dans chaque tronçon sont données dans le tableau 3.

Section	Train A	Train B	Train C
1	3	-	-
2	-	4	-
3	-	-	3
4	8	3	6
5	6	2	5
6	3	-	-
7	-	2	-
8	-	-	2
9	3	2	-
10	2	1	-
11	16	-	-
20	-	20	-

Tableau 2 : Durées de parcours des tronçons selon le train considéré

3.2 Gestion des tronçons conflictuels

Afin d'assurer qu'il n'y ait qu'un seul train à tout instant sur un même tronçon, il est nécessaire d'ajouter des contraintes au graphe caractérisant le parcours des trains. Ainsi, un train ne peut rentrer dans un tronçon commun que lorsque le train qui le précède en est sorti (donc lorsque le train précédent est entré dans le tronçon suivant !). Par exemple, supposons que deux

trains A et B aient à emprunter une voie commune (le tronçon 4), le train A étant prioritaire sur le train B. Une contrainte de séquencement d'une durée égale à la marge de sécurité désirée est insérée dans le graphe pour éviter le conflit (voir figure 9).

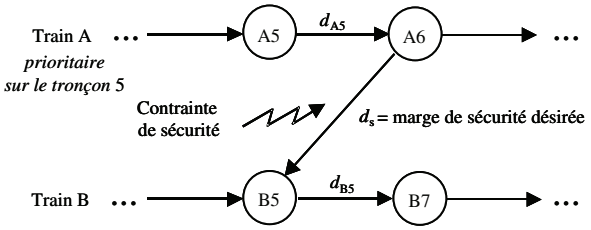

fig 9 : Exemple d'insertion de contrainte pour éviter un conflit sur un tronçon commun

3.3 Travail demandé à l'étudiant en TP

3.3.1 Implantation de l'algorithme de Ford

Avant de pouvoir utiliser OrdoNet, il est demandé aux étudiants d'implanter l'algorithme de Ford dans le fichier **algo.m**. Ce dernier précise les structures de données manipulées et la partie initialisation de l'algorithme. Il doit toutefois être complété à partir du pseudo-code fourni dans le cours et le texte de TP pour que l'application logicielle soit opérationnelle.

3.3.2 Modélisation et ordonnancement des trains sur le réseau

Une fois l'algorithme de calcul implanté, il est demandé aux étudiants de construire et simuler le graphe relatif aux parcours des trains à partir des données exposées précédemment. On précise par ailleurs les ordres de priorité entre les trains :

- Dans la première partie des trajets (départ de la zone de stationnement jusqu'à la prise des wagons), le train B est prioritaire devant le train A lui même prioritaire devant le train C.
- Les trains ne peuvent ressortir de la zone où ils prennent leurs wagons (via le tronçon 4) que s'ils y sont tous entrés. Le train C devient alors prioritaire devant le train A lui même prioritaire devant le train B.

3.3.3 Analyse de modification des contraintes de séquencement

On propose ensuite de modifier la durée des marges de sécurité associées aux contraintes de séquencement. En particulier, on augmente une des contraintes entre deux tâches d'une valeur égale à la marge entre ces deux mêmes tâches, faisant ainsi apparaître plusieurs chemins critiques en parallèle. Les étudiants sont amenés à analyser pour les différentes situations proposées, l'autonomie ou la criticité des tâches. Ils peuvent s'appuyer sur la simulation des graphes et sur les tracés des diagrammes de Gantt correspondants, obtenus à l'aide du logiciel.

3.3.4 Optimisation des parcours

Enfin, on suggère aux étudiants d'améliorer les performances globales du réseau en modifiant les ordres de priorité entre les trains. Il apparaît en effet bien plus judicieux de donner la priorité au train le plus lent (les trains plus rapides pouvant rattraper le retard dû aux temps d'attente devant les tronçons conflictuels) sur l'ensemble du trajet. Cette solution permet de réduire la longueur du chemin critique.

4 RETOUR D'EXPERIENCES

L'ergonomie d'OrdoNet reste certes à améliorer car la saisie du graphe sous forme matricielle dans le fichier texte **matrice_arc.m** est relativement fastidieuse. Toutefois, une fois cette étape franchie, l'interface facilite largement le calcul et l'analyse d'ordonnancements de tâches, à partir des diagrammes de Gantt.

Le retour d'expérience vis-à-vis des étudiants est très positif. Le TP permet de bien comprendre la partie « algorithmique » qui est parfois difficile à assimiler lors d'un cours théorique. A travers l'exemple de la gestion du réseau ferroviaire, il montre clairement les différents niveaux de commande en Automatique et l'intérêt de la planification des activités. Par ailleurs, nous avons pu constater, lors des examens de travaux pratiques réalisés en fin de semestre, que les élèves étaient capables de traiter correctement des sujets comparables (ordonnancement des tâches d'une cellule automatisée, planification des activités d'une ligne de production), maitrisant la modélisation et l'analyse de résultats à partir de l'outil logiciel.

5 CONCLUSION

Dans cet article, nous avons présenté un outil pédagogique développé sous Matlab, permettant la simulation et l'analyse de graphes potentiel-tâche. Cet outil est exploité en travaux pratiques d'Automatique pour illustrer la théorie des graphes et des algorithmes de calcul associés au cours d'une manipulation portant sur l'ordonnancement d'un réseau ferroviaire. Le TP proposé aborde différents aspects liés à la planification de tâches : modélisation, calcul de dates d'exécution, analyse et optimisation.

REFERENCES

[1] M. Gondran, M. Minoux, Graphes et Algorithmes, Eyrolles, 1979
[2] C. Prins, Algorithmes de graphes, Eyrolles, 1994
[3] P. Lopez, P. Esquirol, L'ordonancement, Economica Collection Gestion, Paris, 1999
[4] http://www.GanttProject.org
[5] L. R. Ford and D. R. Fulkerson. Flows in Networks. Princeton Univ. Press, Princeton, NJ,1962.