Quand campus rythme avec ville (part 2)

Marie-Charlotte Tiné

N-SIDE
Chemin du cyclotron 6
B1348 Louvain-la-Neuve – Belgium
marie-charlotte.tine@etu.univ-nantes.fr

Résumé. Dans le cadre de mon projet de fin d'études, j'ai réalisé un stage au sein de l'entreprise N-SIDE en Belgique. J'ai eu l'occasion de participer à différents projets, aussi bien dans le domaine hospitalier que dans celui de l'industrie sidérurgique ou encore dans celui de l'industrie pharmaceutique. Chacun d'eux m'a permis d'accroître mes connaissances en Recherche Opérationnelle et en développement, de perfectionner mon niveau d'anglais ainsi que mes capacités à travailler en collaboration avec d'autres personnes.

1 Remerciements

Il convient, en premier lieu, d'adresser mes remerciements à ceux qui m'ont accueillie et beaucoup appris. Aussi, je remercie Benoît David, mon responsable de stage, qui m'a formée et suivie, ainsi que l'ensemble de l'équipe des consultants de N-SIDE qui m'aura conseillée durant ces quatre derniers mois. Je remercie également Jérome Cornu, Philippe Chevalier et Claire Raucent (N-SIDE) qui ont aidé à la concrétisation du stage, ainsi que l'ensemble de l'équipe enseignante du Master ORO à Nantes.

2 Introduction

De plus en plus d'entreprises souhaitant s'appuyer sur des outils d'aide à la décision font appel à des solutions issues de la Recherche Opérationnelle. L'entreprise N-SIDE s'est spécialisée dans la conception et le développement de tels outils d'aide à la décision et dispose d'une offre de service dans des domaines aussi variés que l'industrie pharmaceutique, l'industrie sidérurgique ou encore dans le secteur de production d'énergie.
Bien qu'N-SIDE soit une société de services avant tout, elle rencontre un réel succès en tant qu'éditeur de logiciels et certains de ses produits sont commercialisés dans plusieurs parties du monde.

C'est en intégrant l'équipe des consultants durant ces quatre derniers mois que j'ai eu l'occasion de travailler sur différents projets. Je présenterai, dans un premier temps, les vocations et activités de l'entreprise et dans un second temps je décrirai le contexte des projets auxquels j'ai participé et mes contributions algorithmique et expérimentale.

3 Présentation de l'Entreprise

N-SIDE a été créée en 2000 comme société dérivée de L'Université Catholique de Louvain-la-Neuve (UCL) par deux professeurs de cette même université: Philippe Chevalier et Yves Pochet, membres des centres de recherches universitaires IAG et CORE. N-SIDE a été conçue afin de répondre aux besoins croissants des entreprises en termes d'optimisation de leurs opérations.
En 2006, le professeur Yves Crama (Service de Recherche Opérationnelle et Gestion de la Production, HEC - ULg) a rejoint l'équipe d'N-SIDE la faisant ainsi évoluer en spin-off conjointe de l'UCL et de l'ULg (Université de Liège).

N-SIDE développe ses activités autour de trois pôles : Modélisation, Optimisation et Distribution de l'information.

Méthodologie de Prise en Charge d'un Projet. De façon assez classique, tout projet débute par une phase d'expression de besoins à l'issue de laquelle l'ensemble de l'information nécessaire à la compréhension du problème est récoltée. À l'issue de cette étape, une collaboration entre les consultants et le client s'engage afin de déterminer les contraintes du problème via une approche logique et mathématique. Sur cette base, les consultants mettent au point un modèle mathématique s'appuyant sur leurs connaissances en Recherche Opérationnelle. Cette étape de prototypage permet de valider les résultats du modèle avec le client qui acquiert à cette occasion une familiarisation avec l'outil qu'il emploiera par la suite et qui réduira ses coûts et/ou augmentera ses profits.

4 Exemple d'Intervention dans le Domaine Pharmaceutique: Optimisation des Essais Cliniques

De plus en plus d'entreprises pharmaceutiques souhaitent optimiser leurs activités cliniques en réduisant les coûts tout en garantissant un haut niveau de service. Ici, dans le cadre d'essais cliniques et afin de tester l'efficacité d'un médicament, on souhaite faire tester des traitements à différents patients. Pour une même étude, les patients se voient en général attribuer un traitement parmi les trois types suivants: un traitement placebo, un traitement servant de comparateur (déjà testé et en place sur le marché) et un traitement contenant le médicament que l'on souhaite tester. On désire donc minimiser la quantité de médicaments à produire tout en assurant une marge minimum et suffisante afin qu'aucun patient n'ait à manquer son traitement.

4.1 CT-FAST

CT-FAST est un outil très puissant permettant, entre autres, de prédire les quantités de productions nécessaires mais également les dates de réapprovisionnement. L'application se compose en trois parties que l'on peut résumer ainsi:

Estimation de la Demande. On calcule, dans un premier temps, une estimation de la quantité de packages [1] dont on aura besoin durant la période d'étude. C'est l'estimation de la demande (voir Fig.1). On prévoit pour cela le nombre de packages nécessaires dans le pire des cas, le nombre de packages nécessaires dans le meilleur des cas ainsi que le nombre de packages nécessaires en moyenne.

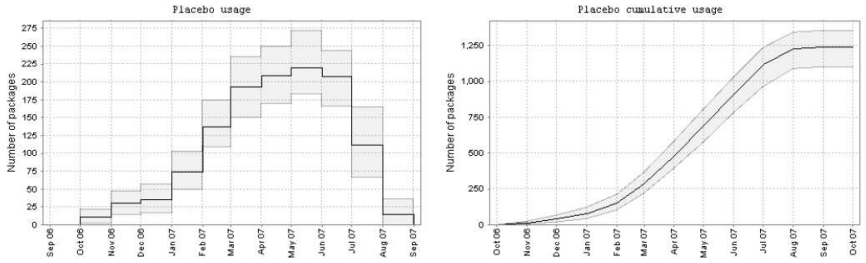

Fig. 1. Estimation de la Demande

[1] Un package représente un médicament dans plusieurs conditionnements possibles. Ex: pilule, fiole, injection...

On peut voir sur la figure 1 l'estimation de la demande au cours du temps. La courbe la plus basse représente le nombre de packages dans le meilleur des cas, la seconde représente le nombre de packages en moyenne et la dernière courbe représente le nombre de packages dans le pire des cas. À chaque point fixé en abscisse correspond en fait une courbe représentant la probabilité de la quantité de packages nécessaires à la date considérée. C'est sur cette courbe que l'on calcule la moyenne ainsi que les valeurs basse et haute.
La courbe de droite reprend les mêmes informations que la courbe de gauche mais les présente en valeurs cumulées.

Estimation du Planning de Production. L'estimation du planning de production consiste à prédire quelle quantité de packages il faudra produire à un moment donné afin de couvrir la demande (Fig.2). On doit tenir compte ici d'un paramètre essentiel: le délai de transport. Pour couvrir la demande à un certain moment, il faut en effet tenir compte du délai de transport des médicaments et donc prévoir assez tôt la commande. Il faut également tenir compte de l'incertitude sur de nombreux éléments, par exemple quand et où [2] arriveront les patients.

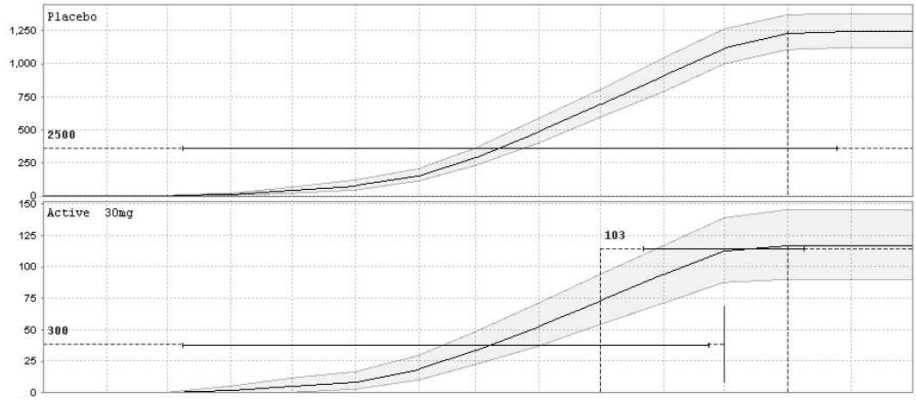

Fig. 2. Estimation du Planning de Production

On peut voir sur la figure 2 l'estimation du planning de production au cours du temps. La courbe la plus basse représente la quantité de packages à produire dans le meilleur des cas, la seconde représente la quantité de packages à produire en moyenne et la dernière courbe représente la quantité de packages à produire

[2] Les patients ont la possibilité de se rendre sur plusieurs sites différents. Chaque site est fourni soit par le dépôt central soit par un dépôt intermédiaire.

dans le pire des cas. La quantité de packages à produire à un moment donné se calcule en faisant la différence entre la quantité requise à cette date et la quantité effectivement produite dans le meilleur des cas au temps d'avant.

Stratégie de Gestion d'Inventaire. La stratégie de gestion d'inventaire (Fig.3) consiste à décider des dates de commande et des quantités de médicaments à commander. On dispose pour cela de la valeur minimale du stock de médicaments que doit posséder chaque dépôt. Lorsque le stock atteint sa valeur minimale, on effectue une commande. Cependant la quantité de médicaments à commander dépend de la stratégie qu'on utilise: la méthode Min/Max consiste à prévoir les productions au début de l'étude et pour tout l'horizon d'étude alors que la méthode Just In Time consiste à attendre la date de commande pour décider de la quantité à produire. Cette dernière méthode est la plus efficace puisqu'on dispose alors de beaucoup plus d'informations qu'au début de l'étude: nombre de patients enrollés, nombre de visites déjà passées et donc quantité de médicaments déjà consommée...

Fig. 3. Stratégie de Gestion d'Inventaire

On observe sur la figure 3 une stratégie d'inventaire Min/Max. La courbe rose représente les valeurs minimales pour le stock et la courbe bleue représente les quantités de médicaments à commander tout au long de l'étude. Dès que le stock atteint la valeur minimale à un moment donné, on commande la quantité donnée par le point correspondant sur la courbe bleue. Le temps que la commande arrive, le stock peut donc descendre au-dessous de sa valeur limite mais ne doit pas atteindre 0. L'inconvénient majeur de cette stratégie est l'incertitude qui pousse à prévoir des productions plus élevées que ce dont on aura réellement besoin.

La réalisation de ces trois étapes est basée sur les méthodes de Monte Carlo: on réalise des simulations d'événements aléatoires puis on se base sur les différents

scenarii pour calculer des statistiques sur des variables pour lesquelles une démarche d'analyse directe n'est pas envisageable. C'est à partir de ces statistiques qu'on pourra réaliser les trois étapes précédemment décrites.

4.2 Contribution

Dans le but d'améliorer la lisibilité des données, j'ai utilisé la librairie Java JFreeChart permettant la génération de graphes.
J'ai d'abord réalisé un graphe des enrollements des patients au cours du temps (Fig.4) avec la possibilité d'afficher le détail des patients enrollés pour chaque pays et/ou le total sur l'ensemble des pays de l'étude. On peut également choisir d'afficher les valeurs cumulées ou non. À noter qu'une modification apportée dans la table des données met directement à jour le graphe.

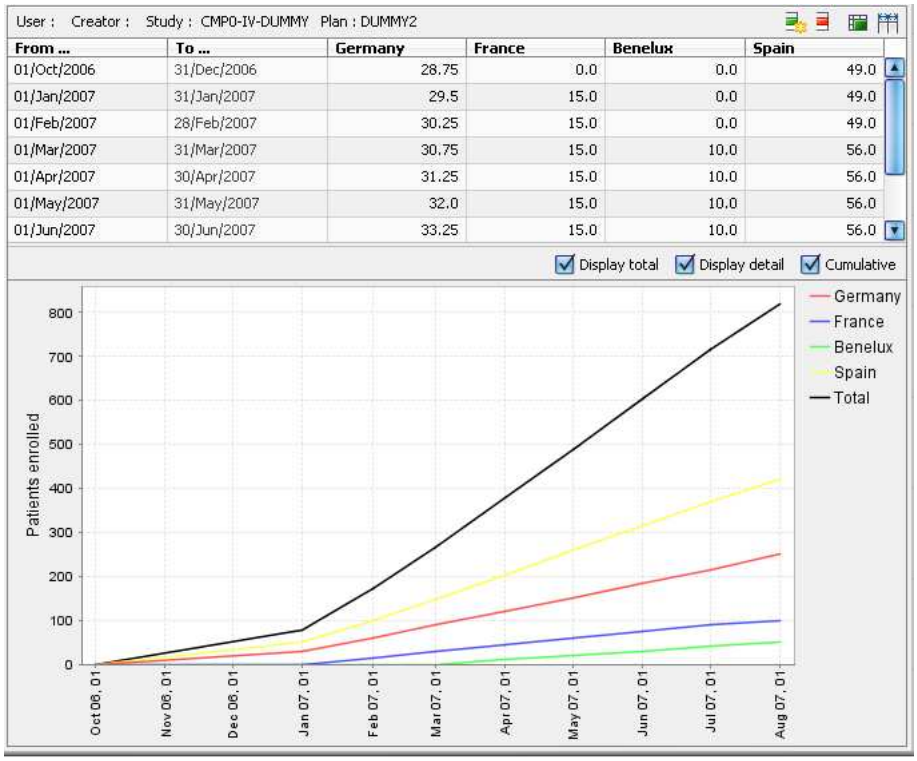

Fig. 4. Graphe des enrollements des patients au cours du temps

J'ai également réalisé un graphique avec sélection interactive (Fig.5): l'utilisateur qui souhaite sélectionner un scénario peut cliquer sur un point de la courbe ce qui aura pour effet, d'une part, de tracer une croix sur le graphe pour qu'il puisse rapidement visualiser les valeurs sur les axes et d'autre part de sélectionner dans la table les lignes correspondantes et de valider le scénario choisi en cochant la case correspondante.

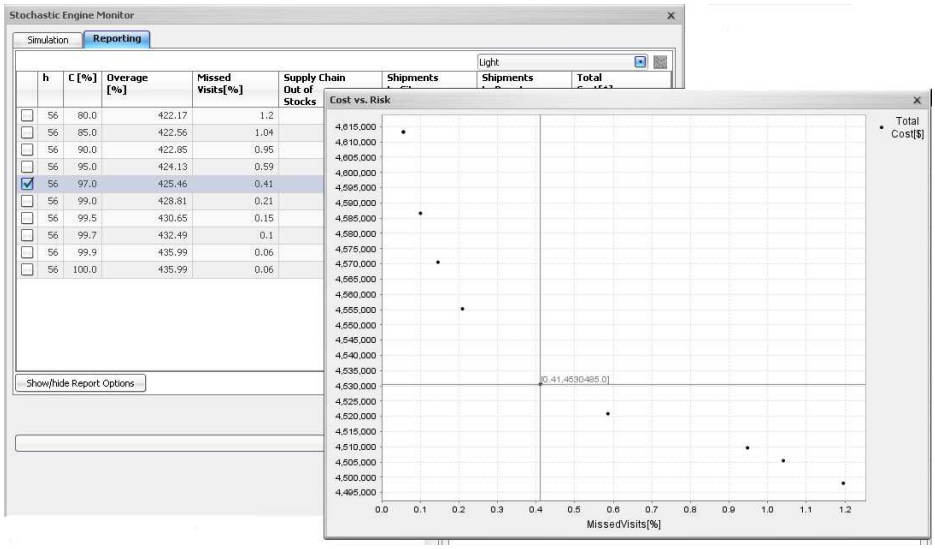

Fig. 5. Graphe avec sélection interactive

Toujours dans un souci de lisibilité, j'ai utilisé la librairie IText permettant la génération de rapports.
L'application CT-FAST permet également la génération de rapports contenant les données initiales et les résultats, sous forme graphique, des trois étapes précédemment décrites. Ces rapports sont lus par l'utilisateur et se doivent donc d'être les plus clairs possible. Cependant, lorsque de nombreux graphes sont générés sur une même page ils s'en trouvent déformés et c'est pourquoi il m'a été demandé d'implémenter une fonction de "split" sur ces tables.

Ci-dessous un aperçu d'une page de rapport avant le split sur les graphes:

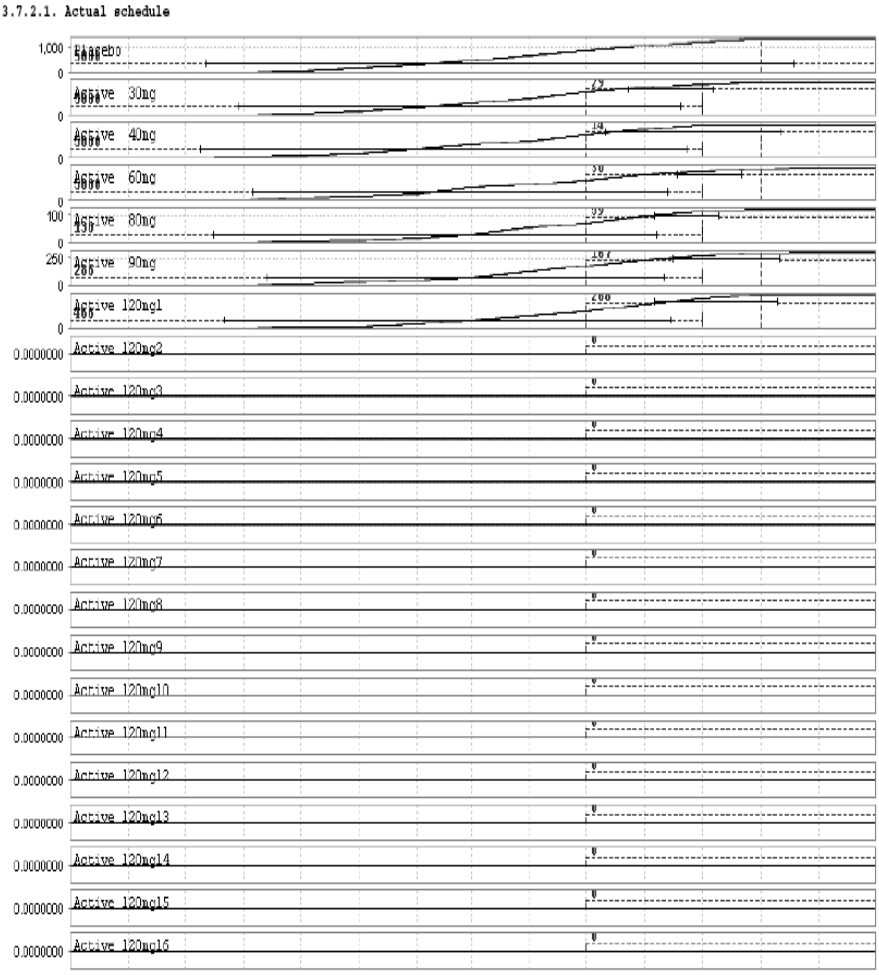

Fig. 6. Graphes avant le split

Et après le split:

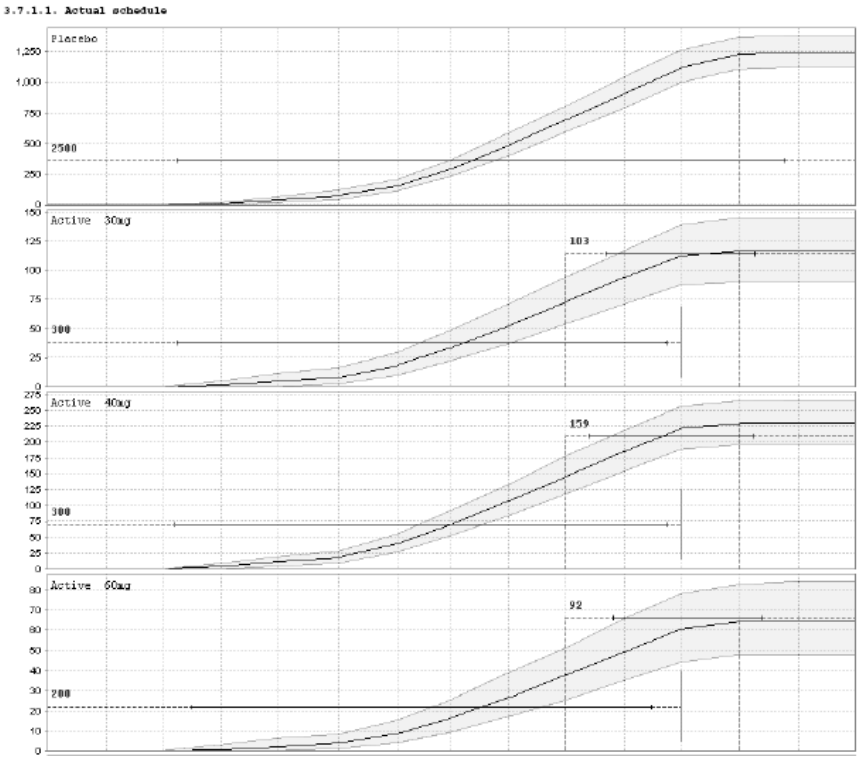

Fig. 7. Graphes après le split

Enfin, afin de permettre une visualisation plus simple et plus claire des données, j'ai réalisé une table regroupant, par date puis par site, les changements sur les courbes de stratégie de gestion d'inventaire. Selon la stratégie, la table contient ou non les valeurs pour chacune des courbes Min et Max.

3.8.5. Chronological changes table

		Placebo		Active 30mg		Active 40mg		Active 60mg		Active 80mg		Active 90mg		Active 120mgl	
Date	WareHouse	Min	Max	Min	Max	Min	Max	Min	Max	Min	Max	Min	Max	Min	Max
08 Aug 06	Depot Spain		936		100		186		53		99		177		328
19 Sep 06	Site Germany-0	6		3		3		3		3		3		4	
	Site Spain-0	4		2		2		2		2		2		3	
	Depot Spain	26		7		9		5		6		10		13	
31 Oct 06	Site Germany-0		20		4		5				4				8
	Site France-0		11		3		3		2		3		3		3
	Site Benelux-0		1												
	Site Spain-0		18		3		5				3		3		7
07 Nov 06	Depot Spain		1174		115		210		66		120		240		441
26 Dec 06	Site Germany-0	12		5		5		4		5		6		8	
	Site France-0	11		4		5		4		4		6		6	
	Site Benelux-0	2		1		1		1		1		1		1	
	Site Spain-0	7		3		4		3		3		4		5	
	Depot Spain	68		13		19		9		13		22		32	
23 Jan 07	Site Germany-0		25										7		
	Site France-0		19										8		8
	Site Benelux-0		5		3		3		3		3		3		4
	Site Spain-0		21												
30 Jan 07	Site Germany-0		39		7		10		5		8		11		17
	Site France-0		32		6		8		5				11		13
	Site Benelux-0		13		4		5				4		4		6
	Site Spain-0		34		5		8		4		5		8		14
13 Feb 07	Depot Spain		740		68		121		41		75		166		299
20 Mar 07	Site Germany-0	16				6						7		10	

Fig. 8. Table chronologique des changements

5 Exemple d'Intervention dans le Domaine de l'Énergie

5.1 COSMOS

Les traders offrent et achètent de l'électricité aux bourses des différents marchés regroupés en deux zones:

- CWE (Central West Europe) comprenant la Belgique, la France, l'Allemagne et les Pays-Bas.
- la Scandinavie regroupant la Norvège, la Suède, la Finlande et une partie du Danemark.

Ils doivent donc acheter ou vendre et décider de l'endroit et de l'heure des échanges afin d'en tirer le plus de bénéfices possible.
Il existe aujourd'hui des lignes électriques au sein de chaque zone (CWE et Scandinavie) mais il existe également des lignes entre ces deux zones. La situation est telle qu'il existe un algorithme (pour décider des quantités d'électricité à faire transiter sur les lignes) pour chaque zone et une enchère pour la capacité de flux d'électricité entre les deux zones. Le résultat en est que parfois l'électricité transite de la zone la plus coûteuse vers la moins coûteuse, ce qui va à l'encontre de la logique puisqu'on aimerait que ce soit la zone la moins coûteuse qui exporte et la plus coûteuse qui importe.
À chaque marché peut être associée une courbe de l'offre et de la demande (voir Fig.10) déterminant les prix sur le marché.
Une étude a donc due être menée par toutes les bourses de ces différents pays pour savoir s'il ne serait pas plus avantageux de n'implémenter qu'un seul algorithme. N-SIDE a réalisé cet algorithme ainsi que des simulations pour pouvoir juger de l'impact d'un tel changement.

Nous pouvons représenter ce problème de flux par un graphe où chaque nœud représente un marché (et donc un prix de vente de l'électricité) et où chaque arête représente une ligne électrique (voir Fig.9). Chaque ligne possède une capacité correspondant au flux maximal d'électricité pouvant transiter.

L'objectif est donc de trouver les quantités optimales d'électricité à faire transiter sur chaque ligne de sorte que chaque marché fasse le plus de profit possible.

La figure 9 illustre un exemple de marchés reliés entre eux par des lignes électriques. Les deux zones sont séparées par la droite noire et les lignes discontinues matérialisent les lignes inter-zones. Le poids sur les arêtes indique la capacité de flux sur la ligne.

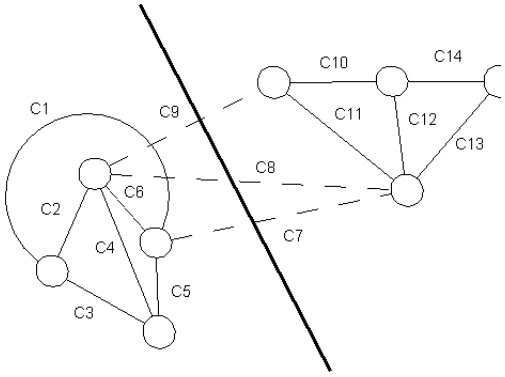

Fig. 9. Exemple de graphe illustrant les différents marchés et les lignes électriques

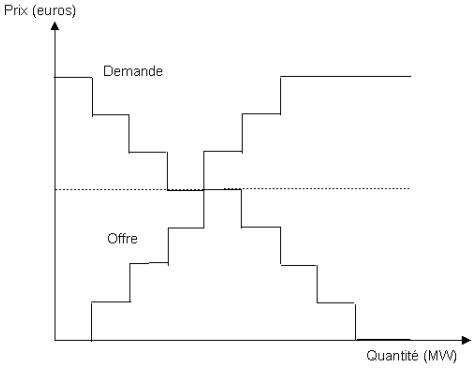

Fig. 10. Courbe de l'offre et de la demande

Comme on peut le constater sur la figure 10 représentant un exemple d'offre et de demande, plus les prix diminuent et plus la demande augmente. De même, plus les prix augmentent et plus l'offre augmente. La ligne discontinue matérialise le fait qu'en dessous de celle-ci les ordres de vente ne sont plus acceptés et les ordres d'achat le sont tous et qu'au-dessus d'elle les ordres de vente sont tous acceptés et les ordres d'achat ne le sont plus.

5.2 Contribution

Une des simulations que l'on souhaite réaliser est la suivante: en fixant le flux sur les lignes propres à chaque zone, on souhaite observer les quantités d'électricité transitant sur les lignes inter-zones afin de juger de l'efficacité de l'algorithme. Après m'être familiarisée avec le logiciel AIMMS, j'ai pu implémenter plusieurs méthodes fixant les valeurs sur les lignes intra-zone d'une part et écrivant ces données dans un fichier d'autre part. C'est après une lecture du fichier de sortie qu'on utilise les valeurs ainsi fixées pour tester l'algorithme.

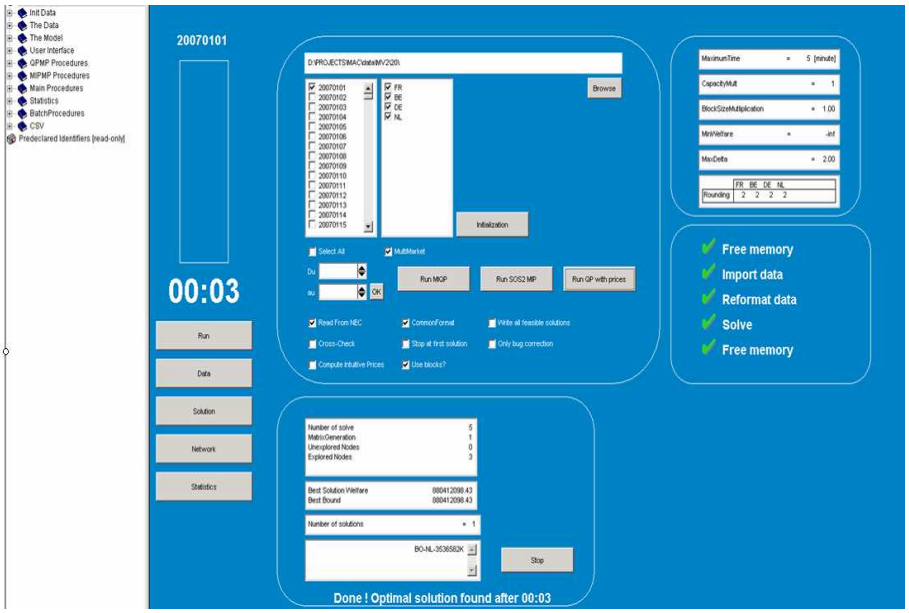

Fig. 11. Aperçu de l'interface de COSMOS

6 Exemple d'Intervention dans le Domaine de l'Industrie de l'Acier

Les coûts en énergie constituent une part importante dans le coût total de la production d'acier. Une solution consiste à réutiliser les différents gaz produits par les cokeries et les haut-fourneaux afin de produire et de négocier de l'électricité. Pourtant produire sa propre électricité n'est pas toujours la solution la plus économique à cause de la volatilité des prix de l'électricité. La quantité d'électricité produite doit être planifiée à l'avance et doit être précise afin d'éviter des pénalités qui réduiraient le rendement.

6.1 SCOOP: Steel Cost Optimization

SCOOP est un outil d'aide à la décision permettant de choisir les quantités et les combinaisons de matériaux à produire afin d'obtenir une solution optimale (de qualité et économique). La particularité de cet outil réside dans le fait qu'il intègre toutes les phases de production de l'acier, là où d'autres outils ne permettent l'optimisation que d'une seule phase en particulier. SCOOP possède donc plusieurs modèles (un pour chaque phase) ainsi qu'un modèle intégré (tous les modèles sont réunis) ce qui offre une vue globale et permet donc une meilleure optimisation.

6.2 Contribution

L'environnement de développement de cet outil s'appuie sur un framework. Un framework est un espace de travail modulaire générique (il est utilisé pour différentes applications) permettant de développer rapidement des applications et offrant une interface graphique facilitant la lecture des données à l'utilisateur. Ici, le framework fait appel au module de résolution (écrit en AMPL [3]) et affiche les résultats de l'optimisation. Une des étapes de l'application consiste, une fois l'optimisation finie, à écrire les variables et leurs valeurs dans un fichier "dump" de sorte que le framework puisse les lire et les afficher dans l'interface. Une des tâches qui m'ont été confiées consistait à tenter de réduire le temps d'écriture des variables dans le fichier, étape qui prenait plus de temps à elle seule que l'ensemble des autres phases, optimisation comprise. En utilisant des commandes AMPL d'affichage des variables du modèle, nous sommes parvenus à diminuer le temps d'écriture d'un facteur dix. J'ai ensuite pu implémenter une méthode Java lisant les données et dont le temps d'exécution est quasiment nul. L'application a ainsi fortement gagné en performances .

Lorsque l'application est lancée, l'utilisateur doit pouvoir l'annuler en cliquant tout simplement sur un bouton "cancel" mais cette fonction n'étant apparemment pas au point, il m'a également été demandé de parer à ce problème.

[3] A Modeling Language for Mathematical Programming.

Pour lancer le programme AMPL à partir de Java, on lance le shell Windows à l'aide de l'exécutable "cmd.exe" à partir duquel on lance le programme en AMPL. Lorsque l'action "cancel" est récupérée, on détruit le processus généré par l'ouverture de l'invite de commande. Ceci a pour effet de fermer l'invite mais ne détruit pas les processus lancés à partir de celle-ci. Par contre, la commande "taskkill" permet, elle, avec les options appropriées, de tuer les processus fils également. Il aura aussi fallu renommer l'invite de commande (on ne peut pas récupérer le PID d'un processus en Java et il faut donc l'arrêter en mentionnant son nom) afin que, pour un même utilisateur, un kill de celle-ci n'ait pas le même effet sur toutes les autres invites. Puisque l'environnement du client peut être multi-utilisateurs, il faut se baser sur le nom du répertoire de travail pour renommer le fichier "cmd.exe" afin qu'un "cancel" provoqué par un utilisateur ne perturbe pas l'environnement des autres utilisateurs.

7 Exemple d'Intervention dans le Domaine Hospitalier

7.1 PlanIsFair

PlanIsFair est un outil d'aide à la décision ayant pour but l'affectation de postes à des assistants. Il assure une répartition équitable des tâches entre assistants en tentant de satisfaire au mieux leurs préférences en termes de postes ainsi que leurs demandes de congés ou encore leurs horaires de garde.

Définition du Problème. L'affectation de postes est valable, ici, pour une période de six semaines qu'on appellera la fenêtre de planning. Un poste se compose d'activités, elles-mêmes définies pour une demi-journée.
Les besoins constituent des tâches, définies sur une demi-journée, qu'il s'agit de couvrir afin d'assurer le bon fonctionnement d'un service. Certaines activités peuvent couvrir un ou plusieurs besoins ou peuvent ne pas en couvrir du tout. Affecter un assistant à une activité pouvant couvrir un besoin signifie que cet assistant couvrira le besoin considéré. Cependant, un assistant ne peut couvrir qu'un seul besoin à un moment donné (une demi-journée), en supposant que son statut le permette.
Il existe trois statuts: Junior 1ère année, Junior et Senior. Le statut agit comme un filtre sur l'affectation possible des assistants aux postes. Plus précisément, tous les postes ne peuvent pas être affectés à tous les assistants. Par exemple, les Juniors de 1ère ne peuvent être affectés à aucun poste en "1ère ligne" car ils ne peuvent remplir aucun besoin fonctionnel "strict" à part certains besoins qui constituent leur seule responsabilité et qu'on leur attribue d'ailleurs en priorité.
Chaque assistant a la possibilité de passer une ou plusieurs demi-journées en consultation. La consultation prime sur l'activité normalement prévue par le poste.
Les assistants définissent, au début de la fenêtre de planning, six préférences en termes de postes. Toutes ces préférences s'équivalent. La satisfaction des préférences représente donc une contrainte pour le problème mais il ne s'agit pas d'une contrainte dure.
Il existe également des besoins en gardes pour la semaine et le week-end. Un assistant de garde la semaine (les soirs) est indisponible pour les activités normalement prévues par son poste. De même, un assistant de garde le week-end est indisponible le lundi pour cause de récupération. Les assistants pouvant être de garde sont connus à l'avance et l'affectation de ceux-ci aux gardes se fait en dehors du modèle lors du post-processing. On essaie de garder une équité dans l'affectation des assistants aux gardes. De plus, les assistants pouvant être de garde ont la possibilité de choisir jusqu'à quatre semaines dites de "non-garde" durant lesquelles on leur assure qu'ils ne seront pas de garde.
Chaque assistant a la possibilité de choisir jusqu'à six semaines de vacances sur la période de planning qui seront considérées par l'application comme une contrainte stricte lors de l'optimisation du planning.

En résumé, il s'agit d'affecter des assistants à des postes en respectant un certain nombre de contraintes. La couverture des besoins et la satisfaction des préférences constituent des contraintes molles dont la minimisation de leur violation constitue la fonction objectif du problème.

7.2 Contribution

Les données étant entrées par les utilisateurs (demande de congés, de non-garde, choix des préférences ...), il faut dans un premier temps vérifier que le problème n'est pas infaisable. Pour cela, j'ai réalisé un test de consistance visant à relever toutes les informations qui pourraient dégrader la solution. Pour ce faire, j'ai implémenté plusieurs méthodes en VBA comptant ainsi le nombre d'erreurs trouvées qui figurera dans un fichier log. La présence d' "erreurs" dans les données initiales n'empêche pas l'optimisation.
Ci-dessous des exemples d'erreurs possibles:

- le nombre d'assistants disponibles à un temps donné est strictement inférieur à l'effectif requis pour couvrir un besoin au même temps
- aucune activité ne peut couvrir un besoin donné
- les activités pouvant couvrir un besoin donné ne figurent dans aucun poste
- un statut ne peut être affecté à aucun poste
- un poste ne peut être affecté à aucun statut

Ces informations permettent donc à l'utilisateur de modifier les données s'il le juge nécessaire.

Le modèle de PlanIsFair étant écrit en Ampl et résolu par le solveur GLPSol, il est intéressant de se demander si nous pourrions améliorer les performances en termes de temps de résolution et/ou de qualité de solution. C'est pourquoi dans un premier temps je me suis penchée sur les éventuels avantages de l'implémentation d'une méta-heuristique. À partir d'une solution initiale générée par un algorithme glouton (implémenté au préalable), on teste les différents échanges possibles (en termes d'affectation de postes aux assistants) en en validant un s'il améliore la solution. On procède ainsi à des échanges successifs jusqu'à ce qu'on ait testé les

$$\sum_{i=1..n} (n-i)$$

combinaisons possibles [4]. On réitère cette opération jusqu'à ce qu'une borne de temps soit atteinte ou qu'il n'y ait plus d'échanges observés.

[4] Où n représente le nombre d'assistants

Malheureusement, en un temps raisonnable, la solution trouvée n'est jamais au moins aussi bonne que celle obtenue par le solveur.
J'ai ensuite eu l'opportunité de travailler sur ce même problème mais en utilisant la programmation par contraintes avec le logiciel COMET [5]. Bien qu'il ait été difficile au début de trouver une modélisation adaptée à la programmation par contraintes, nous avons eu l'occasion d'observer la puissance de cet outil grâce à une étroite collaboration avec la société Dynadec.
Les résultats que nous avons obtenus sont très positifs, voire meilleurs que ceux obtenus avec un Mixed Integer Programming mais il reste cependant difficile de relâcher plusieurs contraintes et c'est pourquoi le modèle de programmation par contraintes ne tient pas compte, pour l'instant, des préférences des assistants.

Pour réaliser ce modèle, nous avons utilisé des contraintes globales telles que atmost [6] pour limiter le nombre d'affectations à un même poste et cardinality [7] pour couvrir au moins les effectifs requis pour la couverture des besoins. Nous avons également employé de la recherche locale afin de guider la construction de l'arbre de recherche: pour un assistant donné, guider la recherche en essayant de commencer à l'affecter à des postes qui lui correspondent (les postes ne contenant aucune activité servant à remplacer les assistants qui seraient absents doivent, par exemple, être affectés en premier à des assistants Seniors).

[5] Pascal Van Hentenryck, Directeur du Laboratoire d'Optimisation de l'Université de Brown et fondateur de la société Dynadec
[6] Système de programmation par contraintes CHIP
[7] Jean-Charles Régin

8 Conclusion

Tout d'abord, ce stage m'a permis de prendre contact avec le monde de l'entreprise. N-Side est une jeune entreprise de petite taille constituée de consultants de haut niveau. De ce fait, j'ai pu avoir accès à des projets touchant à différents secteurs d'activité et être en contact avec leurs concepteurs ce qui a permis des échanges très fructueux.
Le fait d'effectuer ce stage en Belgique m'a également permis de confronter deux univers culturels très proches sur le fond mais présentant des particularismes de formes intéressants.
D'un point de vue de la conduite de projet, j'ai eu l'occasion de visualiser des expressions de besoins rédigées avec les utilisateurs et comprendre ainsi l'importance de cette étape pour la compréhension et la réussite d'un projet.
Les exemples concrets sur lesquels j'ai travaillé m'ont permis de voir la mise en pratique de concepts abordés lors de ma formation à la faculté. Ils m'ont permis également de visualiser quelques problématiques d'optimisation de process dans des environnements aussi variés que ceux de la pharmacie, du milieu hospitalier ou encore de l'énergie.

Ce stage m'a permis également de travailler dans un environnement (framework) de développement spécifique utilisant des outils et des méthodes en rupture avec les habitudes du développement individuel.

Enfin, l'activité d'export de l'entreprise ainsi que la situation spécifique de la Belgique en matière de langues m'ont amenée à perfectionner mon anglais.

References

1. Régin, J-C.: Generalized Arc Consistency for Global Cardinality Constraint. AAAI-96 Portland, OR, USA, pp 209–215 (1996)
2. Van Hentenryck, P., Michel, L.: Constraint-Based Local Search. The MIT Press, (2005)
3. Fourer, R., M. Gay, D., W. Kernighan, B.: AMPL A Modeling Language for Mathematical Programming. Thomson, (2003)
4. Oplobedu, A., Marcovitch, J., Tourbier., Y.: CHARME: Un langage industriel de programmation par contraintes, illustre par une application chez Renault. In Proceedings of the Ninth International Workshop on Expert Systems and their Applications: General Conference, volume 1, pages 5570, (1989)
5. Lowagie, B.: iText in Action Creating and Manipulating PDF Manning Publications, (2006)

A Annexe

Problème des mariages stables étudié lors d'une formation à l'outil COMET à Liège, donné à titre d'exemple.
Définition: si dans un couple, l'homme X préfère une femme Z à sa femme Y et que Z préfère son mari à X ou si, dans ce même couple, Y préfère l'homme W à son mari et que W préfère sa femme à Y, alors il s'agit d'un mariage stable.

```
/*******************************************************************************
  COPYRIGHTS AND TRADEMARKS
  Copyright (c) 2007-2009 Dynamic Decision Technologies, Inc.  All rights reserved.

  Portions of this product are copyright of, and licensed from, Brown University
  or the University of Connecticut.  This product also includes open source software
  developed by the OpenSSL Project, Free Software Foundation, Inc., the Regents
  of the University of California and NetBSD Foundation, Inc.  See copyright notice
  included with the software for additional copyright information.

  DYNADEC TM and COMET TM are trademarks of Dynamic Decision Technologies, Inc.
  *******************************************************************************/

import cotfd;

Solver<CP> CP();

enum Men = {Richard,James,John,Hugh,Greg};
enum Women = {Helen,Tracy,Linda,Sally,Wanda};
range R = 1..5;
//for each of the men, his ranking of the women
Women favWomen[Men,R] = [[Tracy,Linda,Wanda,Sally,Helen],
                         [Tracy,Sally,Linda,Helen,Wanda],
                         [Wanda,Linda,Tracy,Sally,Helen],
                         [Helen,Wanda,Sally,Linda,Tracy],
                         [Sally,Linda,Tracy,Helen,Wanda]];
//for each woman, here ranking of the men
Men favMen[Women,R] = [[Richard,James,Hugh,John,Greg],
                       [John,Hugh,Richard,Greg,James],
                       [Hugh,John,Greg,James,Richard],
                       [Richard,Greg,James,Hugh,John],
                       [Greg,James,John,Richard,Hugh]];

//rankMen[m,w] is the position of woman w in the
//ranking of man m (smaller position is better)
int rankMen[Men,Women];

//ranWoman[w,m] is the position of man m in the
```

```
//ranking of woman w (smaller position is better)
int rankWomen[Women,Men];

forall(m in Men,i in R)
   rankMen[m,favWomen[m,i]] = i;
forall(w in Women,i in R)
   rankWomen[w,favMen[w,i]] = i;

//wife of each men
var<CP>{Women} wife[Men](CP,Women);
//husband of each woman
var<CP>{Men} husband[Women](CP,Men);

solveall<CP> {
//reciprocity of weddings
   forall(i in Men){
//if a man i is married with a woman "wife[i]" then the
//husband of this woman is i
     CP.post(husband[wife[i]] == i);
   }

   forall(i in Women){
//if a woman i is married with a man "husband[i]" then the
//wife of this husband is i
     CP.post(wife[husband[i]] == i);
   }

   forall(i in Men,j in Women){
//if a man i prefers the woman j than his wife, then this
//woman j prefers her usband rather than the man i
     CP.post(rankMen[i,j] < rankMen[i,wife[i]] =>
       rankWomen[j,i] > rankWomen[j,husband[j]]);
   }

   forall(i in Men,j in Women){
//if a woman j prefers the man i than her husband, then this
//man i prefers his wife rather than woman j
     CP.post(rankWomen[j,i] < rankWomen[j,husband[j]] =>
                     rankMen[i,j] > rankMen[i,wife[i]]);
   }

} using {

   label(wife);
   label(husband);
```

```
   show(wife);
}

cout << "Game over!" << endl;
function Setw setw(int n) { return new Setw(n); }
function void  show(var<CP>{Women} []wife) {
   forall(i in Men)
      cout << setw(10) << i;
   cout << endl;
   forall(i in Men)
      cout << setw(10) << wife[i];
   cout << endl;
}
```