Résolution des conflits d’une base chirurgicale

Par Samy Rezig et Yves Tran

Université Paris-Dauphine

Projet Java

**Introduction**

Etant donnée une base de données chirurgicale contenant des informations relatives à chacune des chirurgies, détecter et résoudre les conflits entre opérations revient à analyser les ressources disponibles et utilisables (les chirurgiens et les blocs opératoires), étudier l’ensemble des chirurgies, gérer les ressources et les durées, mettre en place une stratégie de correction et mesurer la qualité de cette dernière. En effet, nous avons à faire à **des erreurs de saisies** qui peuvent avoir lieu sur les horaires des opérations, le nom de la salle et/ou du chirurgien. Ces erreurs peuvent provenir d’une **mauvaise prise d’information** (une mauvaise salle a été inscrite), ou une **absence de valeur** ce qui entraine l’apparition d’une valeur par défaut (par exemple, si l’horaire de début n’est pas saisi, l’heure 8 :00 est mise par défaut). L’objectif de ce projet est de **résoudre un maximum de conflits** tout **en gardant un certain réalisme** au niveau des chirurgies. Pour y parvenir, nous mettrons en place une **analyse pertinente** de la base de données suivi **d’une stratégie adaptée** tout en cherchant à **maximiser une mesure de performance des corrections**.

# La structure du code

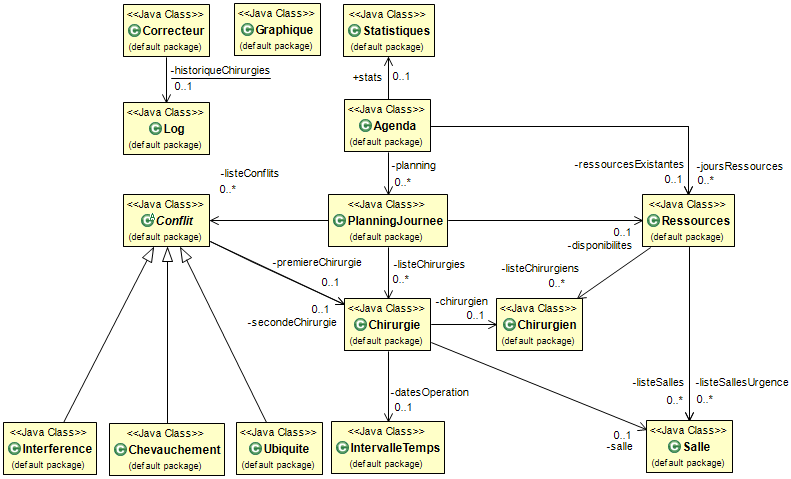


Figure 1 Diagramme de classe

|  |  |
| --- | --- |
| Classes | Description |
| Agenda | Contient le planning (chirurgie et ressources d’un certain jour) et les ressources existantes. |
| Statistiques | Permet de réaliser des statistiques à un moment donné sur un agenda. |
| PlanningJournee | Contient les chirurgies et ressources disponibles d’un certain jour. |
| Chirurgie | Représente une chirurgie (identifiant, chirurgien, salle, les dates de début et de fin). |
| Ressources | Contient des listes de chirurgiens, de salles non urgentes et d’urgence. |
| IntervalleTemps | Représente un intervalle de temps. |
| Salle | Représente une salle de la base de données. |
| Chirurgien | Représente un chirurgien de la base de données. |
| Graphique | Pour afficher un graphique représentant l’évolution du nombre de conflits en fonction du temps. |
| Correcteur | Contient des méthodes statiques utilisées pour résoudre des conflits entre deux chirurgies. |
| Log | Permet de retracer l’historique des différents états des chirurgies. |
| Conflit | Contient deux chirurgies en conflit : un temps qui se chevauche avec au moins une ressource partagée. |
| Chevauchement | Sous classe de conflit, présente deux chirurgies avec un temps qui se chevauche et partage deux ressources communes : le chirurgien et la salle. |
| Interference | Sous classe de conflit, présente deux chirurgies avec un temps qui se chevauche et partage la même salle. |
| Ubiquite | Sous classe de conflit, présente deux chirurgies avec un temps qui se chevauche et partage le même chirurgien. |

# Méthodes de correction de la base de données

1. Hypothèses et méthodes générales de modifications des chirurgies

Ici, nous réalisons **quatre hypothèses**. Nous supposons que **les erreurs de saisie ne se situent pas sur les dates** des chirurgies. Par ailleurs, nous faisons l’hypothèse **qu’il existe un planning de chirurgiens disponibles pour chaque jour** que nous chercherons à déterminer. De plus, nous supposons **que toutes les salles sont disponibles à n’importe quel jour**. Enfin, nous pouvons imaginer **que les chirurgiens possèdent tous un temps de travail par chirurgie équitable**. La même hypothèse est faite pour les salles.

Afin de modifier une chirurgie, il est possible de :

* **Normaliser sa durée** : rendre raccourcir la durée de la chirurgie
* **Modifier une ressource** : changer la salle ou le chirurgien qui interviennent dans la chirurgie
* **Découper une chirurgie** : raccourcir la durée de la chirurgie
* **Décaler une chirurgie** : translater la chirurgie suffisamment loin de celle qu’il intersecte.

Pour **mesurer la cohérence** d’une base de données chirurgicales, nous prenons en compte :

* **L’écart-type des durées moyennes réparties entre les chirurgiens** (1)
* La **durée d’allongement des journées**, il s’agit de la différence entre l’horaire de fin de la dernière chirurgie et l’horaire de début de la première chirurgie de la journée (2)
* Le **nombre de conflits** restant après correction (3)
* Le **nombre de décalages et découpages** effectués (4)
* Le **nombre de conflits corrigés par correction effectuée (noté pertinence)** ; il s’agit du nombre de conflits corrigés par le nombre de corrections total ; sa valeur est généralement comprise entre 0 et 1. (5)

Ainsi, nous cherchons à **minimiser (1) et (2)** pour avoir une répartition cohérente des durée opératoires entre les salles et chirurgiens **ainsi que** **(3) et (5)** pour éviter le **décalage / découpage abusif** des chirurgies. En outre, nous voulons **maximiser (6) qui mesure la pertinence des corrections.** L’idéale est d’avoir au moins une correction qui résout un conflit sans en engendrer par la suite. Dans ce cas, la pertinence est de 1.

1. Analyse de la grande base de données chirurgicales et présentation des outils de correction

* Statistiques générales

|  |  |
| --- | --- |
| Statistiques | Valeurs |
| Durée moyenne des chirurgies | 104 mins |
| Premier quartile des durées des chirurgies | 61 mins |
| Médiane des durées des chirurgies | 91 mins |
| Dernier quartile des durées des chirurgies | 134 mins |
| Ecart-type des durées moyennes en salle | 11.33 mins |
| Ecart-type des durées moyennes des chirurgiens | 23.55 mins |
| Nombres de conflits | 369 |

A partir des quartiles, nous pouvons considérer que **toute chirurgie dont la durée n’est pas comprise entre les quartiles comportent une durée suspecte**. Cependant, la durée ne peut être le seul indicateur de cohérence. On peut également prendre en compte les horaires des chirurgies.

* La normalisation

|  |  |
| --- | --- |
| Horaires | Fréquence d’apparition chez les chirurgies en conflit |
| 08 :00 | 88 |
| 00 :00 | 29 |
| 14 :00 | 15 |
| 13 :40 | 14 |
| 12 :35 | 13 |
| 14 :15 | 12 |
| 15 :15 | 12 |
| … | … |

Il est raisonnable de penser que l’horaire de début et de fin par défaut sont respectivement 8 :00 ou encore 00 :00.

Ainsi, il est donc **important de normaliser les chirurgies trop longues** pour réduire ces fréquences. Nous mettons en place **un outil de normalisation** qui consiste à **repérer les chirurgies trop longues (> dernier quartile)** et **les réduire si ces horaires sont présents dans le tableau**. La durée finale devient le dernier quartile.

* Modification des ressources après analyse des disponibilités des chirurgiens

Nous avons mis à profit **le principe d’héritage** de la programmation orientée objet. La classe Conflit contient une **méthode abstraite pour modifier une ressource**. Les **différentes classes qui étendent cette dernière** peuvent **gérer les ressources comme elles le devraient**. Par exemple, une ubiquité devrait changer le chirurgien dans le conflit.

Nous affichons une partie de l’emploi du temps des chirurgiens en colonne.

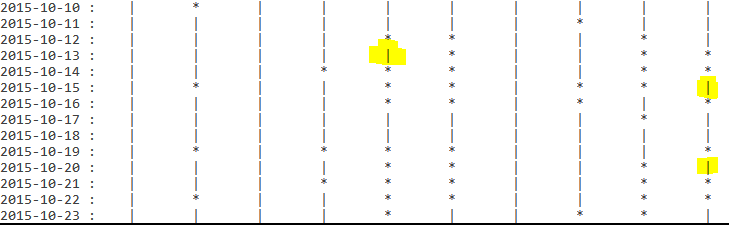


Figure 2 Emploi du temps des chirurgiens

\* le chirurgien a travaillé durant cette journée ;

| le chirurgien n’est pas travaillé

Nous distinguons en jaune des jours où certains **chirurgiens devraient travailler** alors qu’ils sont marqués absents. Ces chirurgiens sont donc disponibles ces jours-ci. Nous allons **compléter l’emploi du temps** en fonction **des fréquences de travail des chirurgiens** pour un certain **jour de la semaine** comme suit :

*POUR TOUT chirurgien*

*Déterminer le nombre de fois qu’il travaille pour un certain jour de la semaine*

*POUR TOUTE semaine*

*SI le chirurgien travaille moins de 5 jours et plus que 1 jour*

*Lui attribué ses jours de travail les plus fréquents pour avoir 5 jours de travail*

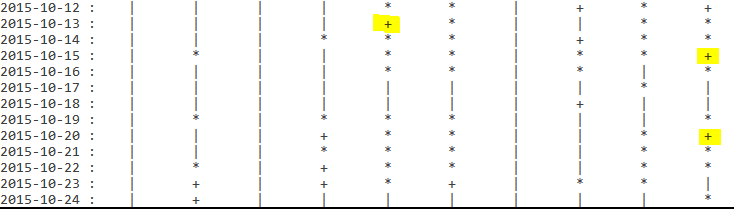


Figure 3 Emploi du temps des chirurgiens complété

Pour **modifier un chirurgien ou une salle**, il suffit de **regarder les disponibilités par jour sans prendre celles utilisées par d’autres chirurgies en même temps.** Ainsi, on définit une liste de **ressources utilisables** comme suit :

*Extraire les listes des ressources disponibles*

*Extraire les listes des ressources utilisées*

*Liste des ressources utilisables = liste des ressources disponibles – ceux utilisées*

La **modification des ressources est possible** si la liste des ressources n’est **pas vide**.

* Découpage d’une chirurgie dans un conflit

Nous décidons de **découper une et unique chirurgie** entre les deux en conflit. Pour **autoriser** le découpage, il faut que les **deux opérations ne commencent ni ne se terminent à des horaires identiques**. Autrement, une chirurgie serait découpée jusqu’à ce que sa durée soit nulle ou trop faible. De ce fait, nous définissons un **taux de superposition** :

Il s’agit du maximum entre la durée d’intersection divisée par la première chirurgie et celle par la seconde. Deux chirurgies complétement superposées donnent un taux égal à 1. Le **découpage est autorisé si le taux est suffisamment petit (< 0.8)** puisqu’il faut prendre en compte la durée inter chirurgicale.

Pour **choisir la chirurgie à découper**, on calcule un **taux de suspections** de chaque chirurgie comme suit :

La chirurgie avec le **taux le plus élevé est choisi** pour être découpée. Elle perd la durée d’intersection avec une durée inter chirurgicale.

* Décalage d’une chirurgie

Le décalage se fait avec **la chirurgie qui commence le plus tard**. Elle est décalée d’une durée *d* où .

1. La stratégie de résolution des conflits

L’idée est de mettre la **méthode de résolution dans une boucle** jusqu’à ce qu’il n’y ait plus de conflits. La méthode est constituée de **règles priorisées** de sorte que la **normalisation** (si besoin) se fait avant de **modifier des ressources** si besoin, qui se fait avant le **découpage** si besoin, qui se fait avant le **décalage** si besoin.

**TANT QU’il** existe des conflits

Mettre à jour la liste des conflits

Trier la liste des ressources disponibles

**POUR TOUT** conflit,

Ordonner les 2 chirurgies par l’heure de commencement

**SI** le conflit existe toujours**, ALORS** normaliser le début de la 1ere chirurgie et la fin de la 2e.

**SI** le conflit existe toujours**, ALORS** modifier les ressources

**SI** le conflit persiste et l’une des chirurgies est trop longue mais pas trop courte

et le taux de superposition < 0.8

**ALORS** calculer les taux de superposition

Choisir la chirurgie avec le taux le plus élevé et la découper

**SI** le conflit persiste, **ALORS** décaler la chirurgie.

1. Les résultats obtenus

Il est possible d’ajouter **des variantes** en triant les listes des ressources utilisables avant de résoudre un conflit.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Pertinence | Itérations | Durée allongement | Nombre de décalages | Ecart-type des chirurgiens |
| Stratégie initiale (sans tri) | 0.9946 | 5 | 0.4628 | 59 | 10.2819 |
| Tri croissant par nombre de chirurgies gérées | 0.8978 | 5 | 0.8946 | 119 | 10.5429 |
| Tri croissant par durée des ressources | 0.8978 | 5 | 0.8946 | 119 | 10.3367 |
| Décalage seulement (pour la pertinence) | 0.5557 | 6 | 12.4216 | 664 | 11.3327 |
| Sans définition des ressources utilisables | 0.2974 | 23+ | 0.4628 | 19 | 10.2638 |

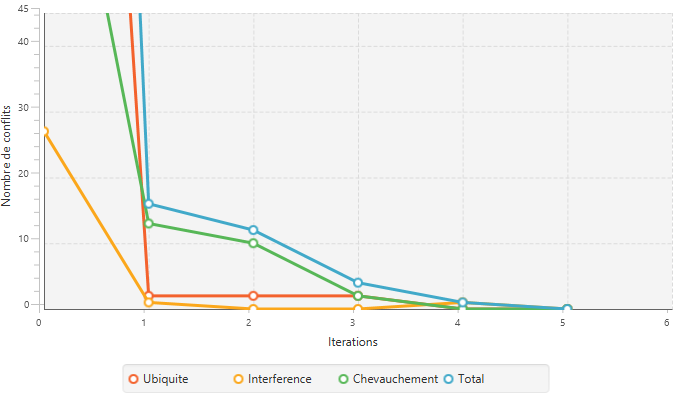


Figure 4 Evolution des conflits à chaque itération avec la stratégie de base (Chirurgies\_v2.csv)

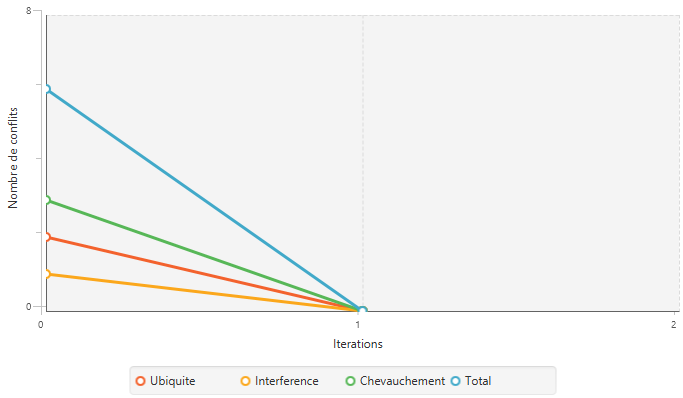


Figure 5 Evolution du nombre de conflits à chaque itération ave la stratégie de base (MiniBase(1).csv)

1. Un retour sur le projet

|  |  |
| --- | --- |
| Utilisation | Localisation dans le code |
| Expressions lambdas | Omniprésence |
| Stream | Omniprésence |
| Collection | Omniprésence |
| Méthode de tri par Java | Classe Ressource et Statistiques |
| Map | Omniprésence |
| JavaFX | Classe Graphique |
| Comparable | Classe Chirurgie |
| Annotation | Omniprésence |
| Lecture/Ecriture de fichiers | Classe Agenda |
| Gestion des erreurs | Classe Main et Agenda |
| Types Enumérés | Classe Chirurgien (DayOfWeek de java.time) |