# Optimisation des tournées

Dans ce chapitre, nous nous intéressons au problème d'optimisation des tournées des ambulances au sein de la Centrale des Ambulanciers. L'étude de ce problème est décomposée en deux parties. Dans chacune d'elles, une approche différente du problème est présentée. Dans la première partie, le problème est abordé dans sa version statique, de manière la plus générique possible. Dans la seconde partie, le problème est étudié en considérant l'ensemble des contraintes spécifiques à la Centrale des Ambulanciers (CA) et dans sa version dynamique. Pour chacune des approches, une modélisation du problème et des méthodes de résolution sont proposées et testées.

## 7.1 Modèle statique du problème des ambulanciers

Dans cette première étude nous avons ramené le problème d'optimisation de tournées d'ambulances à un problème proche du problème du multi-voyageurs de commerce. Nous présentons dans une première section la notation et la modélisation du problème, puis nous proposons deux méthodes de résolution. La première est basée sur la Programmation Linéaire en Nombre Entier (PLNE) en utilisant certaines techniques de pré-processing et de coupes (Kergosien et al. [111]). Nous avons également étendu ce PLNE à un problème de tournée de personnel de soin pour l'hospitalisation à domicile qui ne sera pas détaillé dans ce document ([117] et [114]). Et la deuxième est une heuristique : une recherche tabou avec mémoire adaptative (Kergosien et al. [112]).

### 7.1.1 Définition et modélisation

La première étude aborde le problème dans une version statique. Nous supposons que nous connaissons toutes les demandes de transports à l'avance. Ces demandes sont caractérisées par une heure de prise en charge du patient à son lieu d'origine qui est estimée en fonction de l'heure réelle de rendez-vous du patient au service de destination. Les heures de départ des équipes sont à déterminer mais une durée de travail maximale est à respecter. Nous considérons également que chaque équipe est affectée à un véhicule pour toute la journée. Pour cette première étude, nous ne nous intéressons pas en détail aux différents types

de transports. Cependant, nous considérons que tous les véhicules ne peuvent pas effectuer tous les transports. Cette contrainte permet d'introduire la notion de type de véhicule à utiliser pour un type de transport. Mais elle permet aussi de généraliser le problème en tenant compte par exemple des différents matériels médicaux requis pour certains transports. Etant donné les transports à effectuer dans une journée, l'objectif est de proposer les affectations des véhicules aux demandes et de définir les trajets des véhicules de manière à minimiser les coûts de transports (appel aux ambulances privées et frais kilométriques) et en s'assurant de ne pas faire attendre trop longtemps un patient. Ce dernier critère revient à minimiser le plus grand retard.

#### 7.1.1.1 Notation

Soit $\mathcal{R}$ l'ensemble des $n$ demandes de transports de la journée, chaque demande $i$ est caractérisée par :

- Un lieu de départ $dep_i$ du patient et son lieu de destination $dest_i$,
- Un temps $t_{dep_i,dest_i}$ de déplacement du véhicule entre les points de départ $dep_i$ et de destination $dest_i$,
- Un temps $t_{dest_i,dep_i}$ de déplacement du véhicule entre les points de destination $dest_i$ et de départ $dep_i$,
- Un temps $p_i$ de prise en charge du patient par les ambulanciers pour effectuer le transport en dehors du véhicule (temps pour aller chercher le patient dans le service origine et l'amener au véhicule, et le temps de l'évacuer du véhicule pour l'amener au service de destination), ce temps de prise en charge vient s'additionner au temps de transport.
- Une date $e_i$ de rendez-vous pour le départ du patient de la demande $i$. Le patient ne peut pas être pris en charge par les ambulanciers avant cette date et doit attendre le moins possible l'ambulance qui va effectuer son transport.

Soit $\mathcal{V}$ l'ensemble des $m$ véhicules (ambulances du CHRU et ambulances du privé), chaque véhicule $v$ est caractérisé par :

- Un dépôt $D_v$, on considère qu'un véhicule n'effectue qu'une seule tournée, qu'il quitte son dépôt en début de tournée et n'y revient qu'en fin de tournée. Nous notons également $\mathcal{D}$ l'ensemble des dépôts.
- Une durée maximale $T_v$ d'utilisation du véhicule k avant de revenir au dépôt. Cette durée permet de représenter des contraintes liées à la durée de travail des ambulanciers ou à la limitation du carburant.
- Un coût $c^v_{a,b}$ engendré par l'utilisation du véhicule $v$ pour effectuer le trajet du point $a$ au point $b$. La prise en compte du véhicule dans les coûts se justifie par le fait que les ambulances privées sont plus coûteuses que les ambulances de la Centrale des Ambulanciers. De plus, un coût fixe de mobilisation par véhicule pourra être ajouté à chaque sortie de chaque dépôt.

Chaque patient ne peut pas être transporté par n'importe quel véhicule. Les véhicules présentent des caractéristiques diverses requises pour transporter tel ou tel type de patient. Nous affectons à chaque véhicule $v$ et à chaque demande $i$ un vecteur binaire $S_v$ et $H_i$.

Chaque attribut $s_{v,u}$ de $S_v$ est égal à 1 si le véhicule $v$ possède la caractéristique $u$ et 0 sinon. Et chaque attribut $h_{i,u}$ de $H_i$ est égal à 1 si la demande $i$ nécessite la caractéristique $u$ pour le transport et 0 sinon.

#### 7.1.1.2 Modélisation

Le problème consiste à répondre à toutes les demandes de transports en respectant les contraintes et en minimisant deux critères qui sont le coût total de déplacements de tous les véhicules mobilisés et le temps d'attente maximum des patients avant leur prise en charge. Ce problème est équivalent à un PVC dans un graphe $G$ orienté asymétrique et fortement connexe. Ce graphe est constitué d'un ensemble de sommets divisé en deux sous-ensembles : un sous-ensemble $\mathcal{D}$ de sommets pour tous les dépôts des véhicules, et un sous-ensemble $\mathcal{R}$ de sommets pour toutes les demandes de transports. Un arc entre deux sommets $(i,j) \in \mathcal{R}\times\mathcal{R}$ symbolise le traitement de la demande $i$ et le passage à la demande $j$ comme le montre la figure 7.1. L'arc possède donc :

- Une longueur $d_{i,j}$ égale à $t_{dep_i,dest_i} + p_i + t_{dest_i,dep_j}$.
- Un coût $w^v_{i,j}$ égal à $c^v_{dep_i,dest_i} + c^v_{dest_i,dep_j}$.

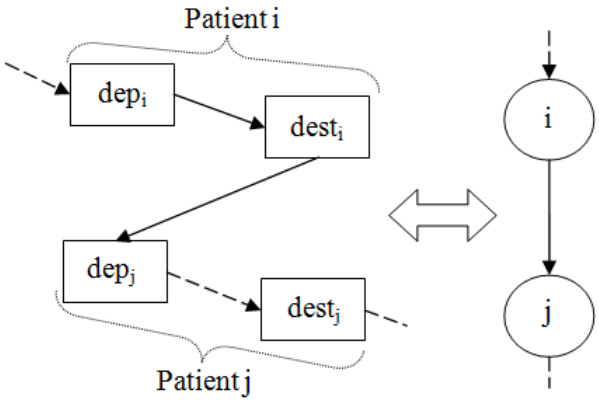


FIG. 7.1 – Modélisation en PVC

Pour les arcs partant d'un sommet dépôt à un sommet représentant une demande, la longueur est simplement égale à la durée pour aller du dépôt au lieu de départ du patient, le coût est égal au coût normal plus un coût fixe de mobilisation. Les arcs de retour au dépôt ont une longueur égale à la durée pour aller du lieu de destination du dernier patient transporté au dépôt.

Les véhicules sont les voyageurs du PVC, ils ont une durée maximale de voyage $T_v$, temps d'attente compris. Tous les voyageurs ne peuvent pas passer par tous les sommets (respect des spécificités des types de véhicules et matériels) et chaque sommet $i$ ne peut être traversé par un voyageur qu'à partir de la date $e_i$. Il y a un seul dépôt par voyageur, et tous les voyageurs ne sont pas obligés de sortir de leur dépôt. Certaines ambulances privées peuvent ne pas être utilisées.

L'objectif de ce PVC est de trouver les itinéraires de chaque voyageur de manière que chaque sommet soit traversé par exactement un seul voyageur tout en respectant les contraintes. Les deux critères à minimiser sont les coûts totaux des trajets des voyageurs et le plus grand retard des voyageurs à chaque sommet. Ce problème multicritère prend en compte aussi bien les coûts de transports pour l'hôpital que la qualité des transports en considérant les retards. L'optimisation multicritère est difficile car il n'existe pas en général de solutions minimisant tous les critères simultanément. Il existe néanmoins plusieurs méthodes pour aborder ce type de problèmes (Steuer, 1986 [183]). La méthode que nous avons choisie est l'approche "epsilon-contrainte" qui consiste à borner un critère par une valeur $\epsilon$ et à minimiser l'autre critère. Ici, nous bornerons le critère du retard maximal. Ceci revient à attribuer des fenêtres de temps $[e_i, e_i + \epsilon]$ pour chaque sommet $i$, de sorte que chaque voyageur doit traverser les sommets à une date comprise entre ces deux valeurs. Toute solution respectant ces fenêtres de temps garantit au patient de ne jamais attendre l'ambulance plus d'un temps $\epsilon$.

### 7.1.2 Résolution exacte (PLNE)

L'une des premières modélisations en PLNE pour le PVC a été formulée par Miller et al. en 1960 [144]. Cette formulation a été étendue par Svetska et Huckfeldt en 1973 [184] à plusieurs voyageurs. Les auteurs proposent une procédure par évaluation et séparation pour résoudre ce problème en utilisant une relaxation du programme linéaire. Dans l'article de Kara et Bektas (2006) [109], les auteurs proposent d'étendre ces modèles en intégrant un nombre minimum de sommets à visiter pour chaque voyageur aussi bien dans le cas mono que multi-dépôts. Nous nous sommes inspirés de ces modèles pour établir un PLNE du problème de transports de patients.

#### 7.1.2.1 Les variables

Etant donnés $\mathcal{R}$ un ensemble de demandes, $\mathcal{D}$ un ensemble de dépôts et $\mathcal{V}$ un ensemble de véhicules, les variables du modèle sont les suivantes (nous considérons un dépôt $D_v$ par véhicule $v$) :

- $\forall (i,j) \in (\mathcal{R} \cup \mathcal{D})^2, i \neq j, \forall v \in \mathcal{V} : x_{i,j}^v = \begin{cases} 1 & \text{si le véhicule } v \text{ passe par l'arc } (i,j) \\ 0 & \text{sinon.} \end{cases}$
- $\forall i \in \mathcal{R} : u_i \in \{1...n_{max}\}$ : cette variable permet d'éliminer les sous-tours d'un même véhicule, la valeur est au moins égale au nombre de sommets, appartenant à $\mathcal{R}$ visités par le véhicule depuis le sommet dépôt sachant que $n_{max}$ est le nombre maximal de sommets qu'un même véhicule peut visiter. Cette valeur est déterminée par un pré-processing des données que nous détaillerons plus loin.
- $\forall i \in \mathcal{R} : z_i \geq 0$ : date de départ du sommet $i$.

#### 7.1.2.2 Les contraintes

Les contraintes du modèle sont les suivantes :

$$\forall v \in \mathcal{V} : \quad \sum_{i \in \mathcal{R}} x^v_{D_v,i} \leq 1 \tag{7.1}$$

$$\forall i \in \mathcal{R} : \quad \sum_{v \in \mathcal{V}} x^v_{D_v,i} + \sum_{v \in \mathcal{V}} \sum_{j \in \mathcal{R}} x^v_{j,i} = 1 \tag{7.2}$$

$$\forall i \in \mathcal{R}, \forall v \in \mathcal{V} : \quad x^v_{D_v,i} + \sum_{j \in \mathcal{R}} x^v_{j,i} = x^v_{i,D_v} + \sum_{j \in \mathcal{R}} x^v_{i,j} \tag{7.3}$$

$$\forall (i,j) \in R^2, i \neq j : \quad u_i - u_j + n_{max}. \sum_{v \in \mathcal{V}} x^v_{i,j} \leq n_{max} - 1 \tag{7.4}$$

$$\forall i \in \mathcal{R} : \quad e_i \leq z_i \tag{7.5}$$

$$\forall i \in \mathcal{R} : \quad z_i - e_i \leq \epsilon \tag{7.6}$$

$$\forall i \in \mathcal{R}, \forall v \in \mathcal{V} : \quad d_{D_v,i}.x^v_{D_v,i} \leq z_i \tag{7.7}$$

$$\forall (i,j) \in \mathcal{R}^2, i \neq j : \quad z_i + d_{i,j} \leq z_j + (1 - \sum_{v \in \mathcal{V}} x^v_{i,j}).HV \tag{7.8}$$

$$\forall (i,j) \in \mathcal{R}^2, i \neq j, \forall v \in \mathcal{V} : \quad z_i + d_{i,D_v} - z_j + d_{D_v,j} - T_v \leq (2 - x^v_{i,D_v} - x^v_{D_v,j}).HV \tag{7.9}$$

Les contraintes 7.1 imposent au plus une seule sortie d'un véhicule de son dépôt. Les contraintes 7.2 expriment le fait que tous les sommets $i$ doivent être traversés exactement une fois, autrement dit que chaque patient doit être transporté. Les contraintes 7.3 assurent la continuité des circuits : lorsqu'un véhicule $v$ entre dans un sommet $i$, il en ressort obligatoirement. Les contraintes 7.4 sont aussi appelées *SEC* dans la littérature pour *Substours Elimination Constraints*. Elles permettent d'éliminer les sous-tours d'un véhicule (sous-circuits) en imposant un ordre croissant strict sur les variables $u_i$ dans l'ordre des sommets visités par un même véhicule. Si deux sommets $i$ et $j$ sont visités dans cet ordre par un même véhicule alors $u_i$ est strictement plus petit que $u_j$, il est ainsi impossible de créer un circuit ne passant pas par le dépôt du véhicule [144]. Même si les contraintes *SEC* ne sont pas indispensables, car les contraintes 7.8 permettent aussi l'élimination des sous-tours, nous les utilisons comme une technique de coupe expliquée plus loin dans cette section. Les contraintes 7.5 et 7.6 bornent la valeur de $z_i$ pour chaque sommet $i$ de telle manière que la date de départ du véhicule soit comprise entre $e_i$ et $e_i + \epsilon$. Les contraintes 7.7 (spécifiques au dépôt) et 7.8 expriment les temps de trajets entre deux sommets (HV est une constante positive supposée infini). Entre deux sommets consécutifs $i$ et $j$ visités par un même véhicule, la date de départ $z_j$ ne peut pas être inférieure à $z_i + d_{i,j}$. Enfin, les dernières contraintes 7.9 imposent à chaque véhicule $v$ de ne pas dépasser son temps de trajet maximal $T_v$.

Pour garantir le respect des contraintes de caractéristiques des véhicules à ne pouvoir desservir que certaines demandes, il suffit pour chaque véhicule $v$ qui ne peut pas assurer la demande $i$ de fixer certaines variables à 0 :

$$\forall i \in \mathcal{R} \cup \mathcal{D} : x^v_{i,j} = x^v_{j,i} = 0$$

Enfin, le dernier élément concerne le calcul de $n_{max}$ qui permet de mieux borner la valeur des $u_i$ et d'utiliser cette valeur pour certaines coupes. Ce calcul est basé sur des graphes $G_v$ représentant la succession des demandes possibles par véhicule $v$. Un graphe $G_v$ est donc construit pour chaque véhicule $v$ de la manière suivante :

- un sommet pour chaque demande que peut assurer le véhicule
- un arc entre un sommet d'une demande $i$ à un sommet d'une demande $j$ si $e_i + d_{i,j} \leq e_j + \epsilon$ (le véhicule peut réaliser successivement ces deux demandes) et avec une distance égale à $max(d_{i,j}, e_j - e_i - \epsilon)$ (temps minimum pour réaliser $i$ puis $j$).

Les sommets sont ensuite triés par rang croissant de sorte qu'aucun sommet de rang $r$ ne possède d'arc vers un sommet d'un rang inférieur ou égal à $r$. Pour le calcul de $n_{max}$ par véhicule ($n_{max,v}$), nous appliquons les étapes suivantes :

- construire un tableau de valeurs contenant chaque distance minimale entre un sommet de rang $r$ et de rang $r+1$, la taille de ce tableau est égale au nombre de rangs moins un,
- trier ce tableau par ordre croissant des valeurs des distances minimales entre chaque rang,
- déterminer le $n_{max,v}$ tel que la somme des $n_{max,v} - 1$ premières valeurs du tableau soit inférieure ou égale à $T_v - d_{i,D_v} - d_{D_v,j}$, et la somme des $n_{max,v}$ premières valeurs soit supérieure à $T_v - d_{i,D_v} - d_{D_v,j}$ avec $d_{D_v,j}$ la plus petite distance entre le dépôt $D_v$ et tous les sommets $j$ du graphe, et $d_{i,D_v}$ la plus petite distance entre tous les sommets $i$ du graphe et le dépôt $D_v$. Le nombre $n_{max,v}$ est une borne supérieure du nombre de demandes que peut assurer le véhicule $v$.
- calculer $n_{max}$ tel que $n_{max} = max_{v \in \mathcal{V}}(n_{max,v})$.

Ce calcul ne permet pas de trouver la valeur optimale de $n_{max}$ mais une bonne borne supérieure en un temps raisonnable.

#### 7.1.2.3 La fonction objectif

Les retards étant bornés, seuls les coûts de transports sont à prendre en compte, la fonction à minimiser est donc la suivante :

$$\sum_{\forall v \in \mathcal{V}} \sum_{\forall i \in \mathcal{R} \cup \mathcal{D}} \sum_{\forall j \in \mathcal{R} \cup \mathcal{D} \setminus i} w^v_{i,j}.x^v_{i,j} \tag{7.10}$$

#### 7.1.2.4 Le pré-processing

Quelques traitements rapides des données en amont de la résolution du problème permettent de fixer certaines variables du modèle. Ces traitements sont les suivants :

- si deux sommets sont trop éloignés en termes de distance et que leurs fenêtres de temps sont trop proches, ils ne peuvent pas être assurés consécutivement par un même véhicule, ou encore si :

$$\forall (i,j) \in \mathcal{R}^2, i \neq j : e_j + \epsilon < e_i + d_{i,j} \; alors \; \forall v \in \mathcal{V} : x^v_{i,j} = 0$$

#### 7.1.2.5 Les coupes

Nous avons ajouté des coupes au modèle de manière à réduire l'espace des solutions. Une première coupe permet de réduire l'intervalle des $u_i$ en fonction des départs des véhicules des dépôts, si le véhicule arrivant au sommet $i$ ne provient pas de son dépôt alors la valeur de $u_i$ est supérieure ou égale à 2 :

$$\forall i \in \mathcal{R} : u_i + \sum_{v \in \mathcal{V}} x^v_{D_v,i} \geq 2 \tag{7.11}$$

De plus, certains sommets ne peuvent pas être traités par le même véhicule. Deux sommets ayant des fenêtres de temps trop éloignées ou au contraire trop proches nécessitent deux véhicules différents.
$\forall (i,j) \in \mathcal{R}^2, i \neq j : \quad Si\ \exists\ v \in \mathcal{V}\ tel\ que\ :$

$$d_{D_v,i} + max(d_{i,j}; e_j - e_i - \epsilon) + d_{j,D_v} > T_v \quad ou \quad (e_j + \epsilon < e_i + d_{i,j}) \wedge (e_i + \epsilon < e_j + d_{j,i})$$

$$Alors \quad \sum_{l \in \mathcal{R} \cup \mathcal{D} \setminus i} x^v_{l,i} + \sum_{l \in \mathcal{R} \cup \mathcal{D} \setminus j} x^v_{l,j} \leq 1 \tag{7.12}$$

Ces deux coupes 7.11 et 7.12 sont dites "légères". Enfin, deux dernières coupes peuvent être déduites des graphes $G_v$. Le véhicule $v$ ne peut pas visiter plus de $n_{max,v}$ sommets :

$$\forall v \in \mathcal{V} : \sum_{(i,j) \in G_v} x^v_{i,j} < n_{max,v} \tag{7.13}$$

La dernière coupe s'appuie sur un calcul de plus longs chemins entre les sommets des graphes $G_v$. Etant donnés deux sommets $i$ et $j$ et un entier $p$, soit $A_{i,j,v,p}$ le plus long chemin reliant $i$ à $j$ passant par au plus $p$ sommets. Si la durée de parcours de ce chemin est supérieure à $T_v - d_{D_v,i} - d_{j,D_v}$ alors le véhicule $v$ ne peut pas sortir de son dépôt, suivre ce chemin et revenir à son dépôt. Il y a donc au moins un arc de ce chemin qui ne sera pas emprunté par le véhicule. Cela peut être traduit par la contrainte 7.14 :

$$\forall (i,j) \in \mathcal{R}^2, i \neq j, v \in \mathcal{V} : \sum_{(l,m) \in A_{i,j,v,p}} x^v_{l,m} < |A_{i,j,v,p}| \tag{7.14}$$

Le calcul des plus longs chemins pour les diverses valeurs de $p$ s'effectue par une adaptation de l'algorithme de Floyd [75] utilisé habituellement pour calculer le plus court chemin.

Les résultats du PLNE ainsi que les performances de ces coupes sont présentés dans la section 7.1.4. Ces résultats serviront de comparaison à ceux donnés par la méthode tabou présenté ci-après.

### 7.1.3 Recherche tabou avec mémoire adaptative dans le cas statique

Il existe de nombreuses méthodes de résolution de problèmes de tournées, proposées dans la littérature. Nous avons décidé de choisir parmi les méthodes les plus performantes, celle qui s'adapterait au mieux à notre problème. La méthode finalement utilisée est basée sur celle décrite dans l'article de Taillard et al. [185]. Le problème considéré est un problème statique de tournées de véhicules avec fenêtres de temps souples (*Vehicle Routing Problem with Soft Time Windows*, noté VRP-STW) : étant donné un ensemble de clients à servir et un ensemble de véhicules localisés à un unique dépôt, l'objectif du problème est de déterminer les routes de chaque véhicule de manière à satisfaire les demandes et à minimiser la distance totale des trajets et une somme pondérée de pénalités de retard. Le problème étudié dans cet article possède des points communs avec notre problème : plusieurs véhicules, des demandes à satisfaire caractérisées par des fenêtres de temps, un temps de service à chaque demande et une fenêtre de temps au dépôt précisant l'heure de début et de fin de tournée pour chaque véhicule. De plus, cette méthode a été reprise par les mêmes auteurs (Gendreau et al. [82]) pour résoudre ce même problème dans sa version dynamique (toutes les demandes ne sont pas connues à l'avance). Cependant, d'autres points diffèrent avec notre problème qui présente plusieurs dépôts, des fenêtres de temps rigides, pas de biens à livrer, une adéquation à respecter entre le patient et le véhicule et une fonction objectif différente.

La méthode est basée sur une recherche tabou avec une mémoire adaptative. Nous allons décrire dans cette section l'adaptation que nous avons réalisée pour le problème de transports de patients, de la méthode décrite dans [185] par Taillard et al.

#### 7.1.3.1 Structure générale

L'algorithme 4, nommé $TSAM_{static}$, présente les grandes étapes de la méthode de résolution du problème. Sachant qu'une solution est un ensemble de routes disjointes, l'idée principale de l'algorithme est la suivante. Une liste de routes réalisables est stockée dans une mémoire adaptative. Une route représente la tournée d'une ambulance. Ces routes proviennent des meilleures solutions précédemment explorées. La première étape consiste à initialiser cette mémoire par des routes provenant de solutions initiales aléatoires. Pendant $J$ itérations, des solutions sont construites à partir de cette mémoire adaptative, sont ensuite améliorées et sont ajoutées à la mémoire adaptative. L'amélioration des solutions consiste à réaliser, pendant $I$ itérations, les opérations suivantes : décomposer l'ensemble des routes d'une solution en deux sous-ensembles, améliorer chaque sous-ensemble indépendamment à l'aide d'une recherche tabou et réunifier les sous-ensembles de manière à obtenir une nouvelle solution. Cette décomposition permet d'obtenir un gain en temps de résolution puisque la recherche tabou ne travaille pas sur toutes les routes à la fois (le voisinage d'une solution est donc moins important). A la fin de l'algorithme, une post-optimisation sur la meilleure solution trouvée est appliquée. Nous allons détailler chaque étape et opération importante de l'algorithme général.

**Algorithme 4** $TSAM_{static}$

1: Initialisation de la mémoire adaptative
2: **Pour** $j = 1$ à $J$ **faire**
3: $S \leftarrow$ solution construite à partir des routes de la mémoire adaptative
4: **Pour** $i = 1$ à $I$ **faire**
5: *Décomposition* : création de deux sous-ensembles de routes $S_1$ et $S_2$ à partir de $S$
6: Appliquer la recherche tabou sur $S_1$ pour obtenir $S'_1$
7: Appliquer la recherche tabou sur $S_2$ pour obtenir $S'_2$
8: *Reconstruction* : rassembler $S'_1$ et $S'_2$ pour créer une nouvelle solution $S'$
9: $S \leftarrow S'$
10: **Fin pour**
11: Ajouter les routes de $S$ à la mémoire adaptative
12: **Fin pour**
13: Post optimisation sur la meilleure solution trouvée

#### 7.1.3.2 La mémoire adaptative

La mémoire adaptative est un ensemble de routes réalisables triées par ordre décroissant de qualité. La qualité d'une route est définie par la valeur de la fonction objectif de la solution dont elle provient. Deux actions importantes dans l'algorithme général font intervenir la mémoire adaptative : ajouter des nouvelles routes d'une solution et construire une solution à partir des routes de la mémoire.

La mémoire contient toujours le même nombre de routes, toutes différentes, provenant des meilleures solutions explorées durant la recherche. Quand de nouvelles routes sont ajoutées, les plus mauvaises sont supprimées. Chaque nouvelle route est insérée à la bonne place dans la mémoire adaptative en fonction de sa qualité. Si une route qui doit être ajoutée, est déjà présente dans la mémoire alors :

- soit elle améliore la qualité de celle présente dans la mémoire et elle est alors déplacée à la bonne place vers le début de la liste en fonction de sa nouvelle qualité,
- soit elle n'améliore pas la qualité de celle présente dans la mémoire et elle n'est donc pas ajoutée.

La construction d'une nouvelle solution à partir de la mémoire adaptative se déroule selon les étapes suivantes :

- Une probabilité commune et fixe de sélection, notée $\rho$, est associée à chaque route.
- Les routes sont considérées dans leur ordre dans la mémoire et une route est sélectionnée si un nombre généré aléatoirement est plus grand que $\rho$.
- Après la sélection d'une route, les autres routes de la mémoire adaptative qui ont au moins une demande en commun avec la route sélectionnée ou qui sont affectées au même véhicule, sont éliminées pour la prochaine sélection.
- Si l'ensemble des routes candidates est vide et qu'il reste des demandes non satisfaites, elles sont insérées dans les routes sélectionnées de la même manière que la procédure d'initialisation des solutions décrites ci-après.

#### 7.1.3.3 Procédure d'initialisation

Cette procédure est utilisée pour construire $Nb_{init}$ solutions initiales différentes. L'initialisation est décomposée en deux grandes étapes :

- Les véhicules sont triés dans une liste dans l'ordre croissant des coûts d'utilisation (les véhicules du CHRU avant les véhicules du privé).
- Les demandes sont traitées dans un ordre arbitraire ; une demande est affectée au premier véhicule de la liste pouvant réaliser la demande à l'heure en respectant toutes les contraintes.

Cette procédure permet la génération d'un nombre varié de routes. Après chaque construction d'une solution, une recherche tabou (décrite ci-après) est appliquée avec un petit nombre d'itérations. Les routes de chaque solution améliorée par la recherche tabou sont ensuite ajoutées à la mémoire adaptative.

#### 7.1.3.4 Diversification

Durant la recherche, une même solution peut être explorée plusieurs fois (entre les lignes 4 et 10). Pour éviter cette redondance, une opération de diversification a été ajoutée à la ligne 3 pour explorer d'autres parties de l'espace des solutions encore pas ou peu explorées. Quand la meilleure solution a été générée plus de trois fois durant les $I$ itérations, l'ordre de sélection des routes dans la mémoire adaptative est inversé en commençant donc par les routes de moins bonne qualité.

#### 7.1.3.5 Décomposition et reconstruction des solutions

A chaque itération $i$, une recherche tabou est utilisée pour améliorer la solution courante $S$. La recherche tabou ne travaille pas sur toutes les routes d'une solution, mais seulement sur un sous-ensemble de routes de manière à diminuer le temps de résolution. L'ensemble des routes $S$ est divisé en deux sous-ensembles disjoints notés $S_1$ et $S_2$. Cette décomposition est réalisée en alternant deux approches différentes : une approche géographique et une approche temporelle. L'approche géographique est basée sur la comparaison des centres de gravité des routes en fonction des points desservis par celles-ci. Alors que l'approche temporelle s'appuie sur les heures des transports des patients des routes. A chaque itération $i$ (ligne 4), la route correspondante au $(i \ mod(m))^{\text{ième}}$ véhicule est sélectionnée. Puis les $m/2$ routes les plus similaires selon l'approche choisie (géographique ou temporelle) sont rassemblées pour former l'ensemble $S_1$. L'ensemble $S_2$ est constitué des routes restantes. Après la décomposition, une recherche tabou est exécutée sur chaque sous-ensemble de routes.

La reconstruction de la solution courante consiste simplement à réaliser l'union des deux sous-ensembles des routes améliorées par la recherche tabou.

#### 7.1.3.6 Post optimisation

A la fin de l'algorithme (ligne 13), une post optimisation est appliquée sur la meilleure solution trouvée. Cette opération utilise le même processus qu'à l'initialisation : appliquer la recherche tabou avec un faible nombre d'itérations. Cette optimisation, similaire à l'utilisation d'une recherche locale, est souvent efficace pour améliorer la solution car la recherche tabou travaille sur toutes les routes simultanément et non sur deux sous-ensembles de routes.

#### 7.1.3.7 Recherche tabou

L'algorithme tabou que nous avons utilisé est classique (Glover 1986 [88] et Hansen 1986 [95]) et possède la même structure que dans l'article de Taillard et al. [185]. La recherche tabou est exécutée deux fois par itération $i$ sur deux sous-ensembles de routes. L'algorithme 5 décrit la structure de la recherche tabou implémentée.

**Algorithme 5** Méthode tabou

1: **Tant que** le critère d'arrêt n'est pas atteint **faire**
2: Générer le voisinage de la solution courante en appliquant l'opérateur *"CROSS exchange"*
3: Sélectionner la meilleure solution, non tabou, dans l'ensemble des solutions voisines
4: **Si** la solution est meilleure que la meilleure solution connue **alors**
5: Sauver cette solution en tant que meilleure solution connue
6: **Fin si**
7: Mettre à jour la liste tabou
8: **Fin Tant que**

Le critère d'arrêt de la recherche tabou est un nombre fixe d'itérations défini par la relation suivante :

$$A \times \left(1 + \frac{i-1}{B}\right)$$

$A$ et $B$ sont deux paramètres, et $i$ est l'itération courante des phases de décomposition et reconstruction des ensembles des routes d'une solution (lignes 4 à 10 de l'algorithme $TSAM_{static}$). Le critère d'arrêt dépend de ce numéro d'itération car à chaque exécution de la recherche tabou la qualité des solutions initiales est améliorée. Ainsi la recherche tabou nécessite plus de temps pour s'échapper d'un éventuel optimum local.

Le voisinage de la solution courante est déterminé en utilisant l'opérateur *CROSS exchange*. Comme le citent les auteurs de [185], cet opérateur est particulièrement bien adapté pour des problèmes de tournées de véhicules avec fenêtres de temps, cependant il nécessite une adaptation pour notre problème (tous les véhicules ne peuvent pas transporter tous les patients). Cet opérateur est aussi une généralisation d'opérateurs connus sous les noms de *2-opt* ou *Or-opt*. Il consiste à échanger deux sous-segments entre deux routes comme le présente la figure 7.2.

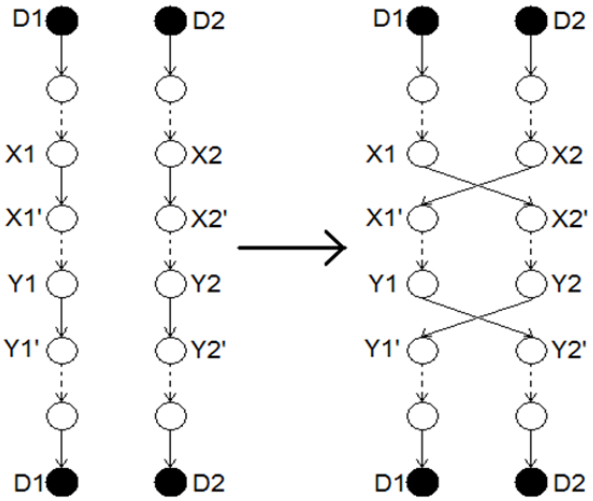


FIG. 7.2 – CROSS exchange

Sur la figure 7.2, $X1'$ est le successeur direct de $X1$, $X2'$ de $X2$, $Y1'$ de $Y1$ et $Y2'$ de $Y2$. L'algorithme 6 décrit la méthode générale utilisée pour calculer l'ensemble des solutions du voisinage entre deux routes uniquement. Les segments $[X1';Y1]$ et $[X2';Y2]$ ne contiennent pas forcément le même nombre de sommets.

L'opération la plus coûteuse en temps dans un algorithme de recherche tabou classique est l'énumération exhaustive de toutes les solutions du voisinage. Nous utilisons une condition nécessaire pour éviter d'énumérer des solutions non réalisables dans le voisinage.

**Condition 1 :**

$$(e_{X2'} + \epsilon \geq e_{X1} + d_{X1,X2'}) \ \wedge \ (e_{X1'} + \epsilon \geq e_{X2} + d_{X2,X1'}) \ \wedge \ (e_{Y2'} + \epsilon \geq e_{Y1} + d_{Y1,Y2'})$$

$$\wedge \ (e_{Y1'} + \epsilon \geq e_{Y2} + d_{Y2,Y1'})$$

Cette condition vérifie si le véhicule 1 a assez de temps pour traiter la demande correspondant au sommet $X1$ puis la demande du sommet $X2'$, mais aussi pour traiter la demande correspondant au sommet $Y2$ puis la demande du sommet $Y1'$. Cette condition effectue la même vérification pour le véhicule 2 (du sommet $X2$ à $X1'$, et du sommet $Y1$ à $Y2'$ respectivement). Etant donné que les fenêtres de temps de notre problème sont de largeur étroites et constantes, cette condition permet d'éviter d'énumérer un grand nombre de solutions non réalisables.

De plus, la valeur de la fonction objectif d'une solution du voisinage peut être calculée sans créer réellement la solution (échanger les sous-segments). Pour améliorer le temps d'énumération des solutions du voisinage, seules les solutions améliorant la meilleure solution du voisinage sont construites. L'évaluation d'un mouvement est calculée de la manière

**Algorithme 6** CROSS exchange

```
1: Pour tous les sous-segments [X1', Y1] de la route du véhicule 1 faire
2:   Si [X1', Y1] contient seulement des transports que peut assurer le véhicule 2 alors
3:     Pour tous les sous-segments [X2', Y2] de la route du véhicule 2 faire
4:       Si [X2', Y2] contient seulement des transports que peut assurer le véhicule 1 alors
5:         Si la condition 1 est validée et l'échange améliore la meilleure solution connue dans le voisinage alors
6:           Echanger les deux sous-segments entre les deux routes
7:           Si la nouvelle solution est réalisable alors
8:             Sauver cette solution en tant que meilleure solution connue dans le voisinage
9:           Fin si
10:        Fin si
11:      Fin si
12:    Fin pour
13:  Fin si
14: Fin pour
```

suivante :

$$f(S') = f(S) - \sum_{x=X1}^{Y1} w^1_{x,succ(x)} - \sum_{x=X2}^{Y2} w^2_{x,succ(x)} + \sum_{x=X1'}^{pred(Y1)} w^2_{x,succ(x)} + \sum_{x=X2'}^{pred(Y2)} w^1_{x,succ(x)}$$

$$+w^1_{X1,X2'} + w^2_{X2,X1'} + w^1_{Y2,Y1'} + w^2_{Y1,Y2'}$$

Où $succ(x)$ et $pred(x)$ désignent respectivement les sommets successeur et prédécesseur du sommet $x$ avant échange, $f(S)$ la valeur de la fonction objectif de la solution S, et $f(S')$ la valeur de la solution voisine.

Ces conditions permettent de réduire considérablement l'ensemble de solutions voisines générées entre deux routes. Pourtant, une solution obtenue après un mouvement et respectant toutes ces conditions n'est pas nécessairement réalisable à cause des fenêtres de temps des demandes. De plus, chaque véhicule possède un temps maximal de trajet qui implique de vérifier si les deux nouvelles routes sont réalisables ou non.

La liste tabou est aussi reprise de l'article de Taillard et al. [185]. Elle est basée sur la valeur de la fonction objectif d'une solution qui est stockée durant un nombre fixe d'itérations égal à la moitié du nombre maximal d'itérations de la recherche tabou. Une solution est tabou si sa valeur de fonction objectif est contenue dans la liste tabou. La taille de la liste tabou est supposée suffisamment grande pour être considérée de taille infinie. Contrairement aux recherches tabou classiques, la liste tabou n'est pas basée sur la représentation d'une solution ou le mouvement pour l'obtenir, car la probabilité pour obtenir deux différentes solutions avec la même valeur de fonction objectif est faible. Cette faible probabilité

s'explique par le grand nombre de valeurs différentes prises par les $w_{i,j}^k$ du fait qu'ils dépendent des frais kilométriques et des types de véhicules.

### 7.1.4 Résultats expérimentaux

Avant de tester les deux premiers algorithmes de la version statique du problème, nous avons généré aléatoirement des instances respectant les hypothèses de départ (cf. section 7.1.1) et dont les caractéristiques sont inspirées de l'activité réelle de la Centrale des Ambulanciers. Le langage utilisé pour l'implémentation est le C++. L'ordinateur utilisé est un Pentium(R) 4 CPU 3.00 GHz, 1.00 Go de RAM et le solveur utilisé pour le PLNE est CPLEX version 10.

#### 7.1.4.1 Génération d'instances

Les jeux de données générés pour les tests possédent les caractéristiques communes suivantes :

- La période est d'une journée, durant laquelle les $m$ ambulances ne peuvent pas rouler plus de 6 à 8 heures.
- Les $m_{pub}$ ambulances publiques (appartenant à la Centrale des Ambulanciers) partent d'un même dépôt, Trousseau, alors que les $m_{priv}$ ambulances privées sont réparties géographiquement (1 à 3 ambulances privées peuvent être regroupées par dépôt), sachant que les ambulances privées coûtent plus cher que les ambulances publiques (un coût fixe d'immobilisation par sortie de dépôt en plus du coût kilométrique).
- Le nombre de caractéristiques a été fixé à 3. Chaque caractéristique a une probabilité de 0,7 d'être présente dans un véhicule donné, qu'il soit du CHRU ou d'une société privée. Chaque véhicule possède donc de 0 à 3 caractéristiques. De la même manière, chaque caractéristique a une chance sur deux d'être requise pour une demande $i$ de transport donné.
- Les $n$ demandes de transports ont au moins un point de départ ou d'arrivée dans un des quatre principaux hôpitaux de Tours. Les points de départs ou d'arrivées sont générés en fonction de probabilité calculées en fonction des transports réels (1/4 pour Trousseau, 1/5 pour Bretonneau, 3/20 pour Clocheville, 1/10 pour l'Ermitage, et 3/10 pour une localisation générée aléatoirement).
- Les heures de demandes sont générées suivant deux lois normales (une pour le matin $\mathcal{N}(10h00, 1h00)$ et une pour l'après midi $\mathcal{N}(16h30, 1h30)$) de manière à prendre en compte les pics de demandes constatés dans chaque demi-journée. Chaque transport a une chance sur deux d'apparaître le matin ou l'après midi.
- Le temps d'attente des patients est borné à $\epsilon = 15$ minutes, et la durée de prise en charge du patient par les ambulanciers hors véhicules varie uniformément entre 3 et 10 minutes.

#### 7.1.4.2 Résultats pour le PLNE

Le PLNE a été testé sur cinq types de jeux de données différents comportant chacun 200 instances. Un type est simplement défini par un couple ($n$,$m$) soit (nombre de demandes, nombre de véhicules). Les tableaux 7.1 et 7.2 présentent le temps de résolution moyen (en secondes) d'une instance et le pourcentage d'instances résolues en moins d'une heure et demie par le PLNE sans coupe et sans pré-processing, puis par le PLNE avec pré-processing et les 2 premières coupes (dites "légères") 7.11 et 7.12 et enfin par le PLNE avec toutes les améliorations proposées (pré-processing et toutes les coupes). Ceci afin d'évaluer l'apport de ces techniques d'amélioration. Les coupes 7.11 et 7.12 sont dites "légères" car elles sont moins nombreuses et impliquent moins de variables que les coupes 7.13 et 7.14 qui mettent en jeu un algorithme de Floyd de calcul de plus long chemin sur un graphe. Le premier tableau montre le temps mis en moyenne par le solveur pour trouver la solution optimale avec les différentes techniques de coupes sans compter les instances non résolues. Cette résolution permet de trouver la meilleure solution en terme de coût pour l'hôpital tout en respectant une qualité de transport qui interdit des retards de plus de $\epsilon$.

| $n$ ; $m$ | PLNE brut | PLNE avec coupes légères | PLNE avec toutes les coupes |
|---|---|---|---|
| 10 ; 5 | 0,06s | 0,03s | 0,03s |
| 20 ; 10 | 60,29s | 1,88s | 2,01s |
| 30 ; 15 | 846,68s | 129,06s | 150,27s |
| 40 ; 20 | - | 887,5s | 853,7s |
| 50 ; 25 | - | 1971,3s | 1707,7s |

TAB. 7.1 – Temps moyen de résolution pour le PLNE

| $n$ ; $m$ | PLNE brut | PLNE avec coupes légères | PLNE avec toutes les coupes |
|---|---|---|---|
| 10 ; 5 | 100,0% | 100,0% | 100,0% |
| 20 ; 10 | 99,0% | 100,0% | 100,0% |
| 30 ; 15 | 55,5% | 99,5% | 100,0% |
| 40 ; 20 | 0,0% | 65,5% | 67,5% |
| 50 ; 25 | 0,0% | 17,5% | 18,5% |

TAB. 7.2 – Pourcentage d'instances résolues par le PLNE en moins de 1h30

Au-delà de 50 demandes et 25 véhicules, peu d'instances sont résolues en moins d'une heure et demie de calcul. Les résolutions du PLNE avec coupes légères et avec toutes les coupes présentent un temps de résolution nettement inférieur à celui du PLNE brut, cependant l'apport des contraintes 7.13 et 7.14 en gain de temps semble minime. Les mêmes remarques peuvent être faites concernant le pourcentage des instances résolues. Cette résolution optimale implique un temps trop important pour être utilisée par la Centrale des Ambulanciers d'autant plus qu'elle utilise un solveur commercial, pour le suivi des véhicules et l'aide à la planification temps réel (réalisée par un régulateur). Cependant ces résultats

vont permettent d'évaluer la méthode tabou.

#### 7.1.4.3 Résultats pour la Recherche tabou avec mémoire adaptative

Après plusieurs expérimentations préliminaires, les paramètres de l'algorithme ont été fixés aux valeurs suivantes (certaines sont aussi reprises de l'article de Taillard et al. [185]) :

- un nombre initial de solutions $Nb_{init}$ égal à 20,
- une taille de la mémoire adaptative égale à $50 \times m$,
- des nombres d'itérations $J$ égal à 50 et $I$ à $m$,
- le calcul du nombre d'itérations de la tabou s'effectue avec $A$ égal à 50 et $B$ à 10,
- une probabilité de sélection dans la mémoire adaptative $\rho$ égale à $1 - 3/$(le nombre de routes dans la mémoire).

Pour pouvoir tester l'algorithme sur de plus grandes instances que celles générées précédemment pour tester le PLNE, de nouveaux jeux de données ont été créés en faisant varier le nombre de demandes et le nombre de véhicules (cent instances par couple ($n$,$m$)). Nous avons tout d'abord comparé l'algorithme de la recherche tabou avec mémoire adaptative avec le PLNE (pré-processing et coupes légères) sur des ordinateurs possédant les mêmes caractéristiques que précédemment. Pour les instances strictement plus grandes que 40 demandes, le temps de résolution de Cplex a été borné (cf. tableau 7.3). Ce tableau 7.3 présente le temps moyen de résolution des deux méthodes ainsi que l'écart relatif moyen $\Delta$, sachant que l'écart relatif ($\delta$) pour une instance est défini par la relation suivante :

$$\delta = 100 \times \frac{f(S_{TS}) - f(S_{PLNE})}{MAX(f(S_{TS}); f(S_{PLNE}))}$$

Avec $f(S_{TS})$ la valeur de la fonction objectif de la meilleure solution trouvée par la recherche tabou avec la mémoire adaptative, et $f(S_{PLNE})$ la valeur de la fonction objectif de la solution optimale trouvée par Cplex pour les instances de moins de 40 demandes comprises ou de la meilleure solution trouvée dans le cas contraire (lorsque le temps de résolution du solveur est limité).

| $n$ ; $m$ | PLNE (Cplex) | Recherche tabou avec mémoire adaptative | |
|---|---|---|---|
| | CPU(s) | $\Delta$(%) | CPU(s) |
| 10 ; 5 | 0,05 | 0,43 | 0,03 |
| 20 ; 10 | 1,80 | 0,41 | 0,80 |
| 30 ; 15 | 165 | 0,64 | 3,20 |
| 40 ; 20 | 4291 | 1,67 | 6,30 |
| 60 ; 30 | 200* | -16,20 | 26,20 |
| 70 ; 35 | 300* | -20,23 | 48,01 |

TAB. 7.3 – Comparaison des deux méthodes

* : indique la valeur du temps de résolution à laquelle le solveur est limité, la solution retournée n'est donc pas nécessairement optimale.

Nous pouvons remarquer que non seulement la recherche tabou avec mémoire adaptative trouve des solutions très proches de l'optimal mais aussi qu'elle nécessite un temps de résolution nettement inférieur pour les petites instances (plus petites que 40 demandes). Pour les autres types d'instances, la méthode trouve des meilleures solutions que Cplex en un temps très raisonnable.

Le tableau 7.4 présente l'amélioration de la meilleure solution initiale par la recherche tabou avec mémoire adaptative sur des instances plus importantes. L'écart relatif moyen $\Delta'$ est calculé de la même manière que précédemment entre la meilleure des $Nb_{init}$ solutions initiales et la meilleure solution trouvée par la méthode.

| $n$ ; $m$ | Recherche tabou avec mémoire adaptative | |
|---|---|---|
| | $\Delta'$(%) | CPU(s) |
| 100 ; 50 | 31,0 | 174,94 |
| 150 ; 75 | 33,4 | 855,11 |

TAB. 7.4 – Amélioration de la meilleure solution initiale

Le tableau 7.4 montre une amélioration moyenne de plus de 30% de la meilleure solution initiale quel que soit le jeu de données, ce qui prouve l'intérêt de l'algorithme. Toutefois, en raison du plus grand nombre de demandes et de véhicules, le temps de calcul est plus long que dans le tableau 7.3.

### 7.1.5 Conclusion

Les résultats préliminaires obtenus pour la version statique du problème de transports de patients ont été encourageants. En effet, l'heuristique mise en place permet d'obtenir en un temps raisonnable des solutions de bonnes qualités. De plus, dans une version dynamique du problème, les instances à résoudre à un instant $t$ seront largement inférieures à 130 demandes (moyenne des demandes par jour à la Centrale). Enfin, la particularité de la mémoire adaptive de stocker un grand nombre de routes différentes permettrait d'obtenir une certaine robustesse de la méthode dans le cas dynamique suite à des événements perturbateurs, c'est-à-dire retrouver des solutions de bonne qualité rapidement. Cependant, outre le fait que certaines hypothèses ont été posées pour le cas statique (durée maximale de travail, pas de changement de véhicules pour les équipes et distinction des différents types de transports par un vecteur binaire), l'inconvénient de cette résolution est la prise en compte des routes des ambulances des compagnies privées. Celles-ci ne sont pas gérées par la Centrale des Ambulanciers, et elles assurent généralement peu de demandes. Il est donc inapproprié de considérer ces routes dans la résolution. Cet inconvénient a été rectifié dans la résolution du problème dynamique en plus de prendre en compte toutes les contraintes de départ.

## 7.2 Modéle dynamique intégrant des contraintes spécifiques aux transports hospitaliers de patients

Dans cette deuxième étude, nous considérons le problème dynamique d'optimisation de tournées d'ambulances avec l'ensemble des contraintes spécifiques à la Centrale des Ambulanciers. Nous définirons dans une première partie le problème complet et la modélisation d'une solution. Nous présenterons dans la deuxième partie, l'heuristique mise en place pour résoudre ce problème dynamique. Cette heuristique est une adaptation et une amélioration de l'heuristique précédente : la recherche tabou avec mémoire adaptative ([110], [120] et [121]). Des résultats expérimentaux sont présentés dans une dernière partie.

### 7.2.1 Définition et modélisation

Contrairement à ce qui a été supposé dans l'étude précédente, on considère maintenant que des demandes de transports arrivent également en temps réel et que seulement 30% des demandes en moyenne sont connues en début de journée. Ces demandes doivent donc être affectées aux ambulances du CHRU en fonction de leurs caractéristiques mais aussi de l'état de la flotte de véhicules à l'instant $t$ (disponibilités, situations géographiques, etc.). Les demandes peuvent être aussi dans ce cas traitées par des ambulances du privé avec un coût plus élevé que si elles avaient été réalisées par celles du CHRU, mais les routes de ces ambulances ne seront pas gérées. Seuls les trajets des véhicules du CHRU seront pris en compte, les demandes non assurées par ces derniers induiront simplement un coût supplémentaire du au fait qu'elles devront être sous-traitées, ce qui dégradera la qualité de la solution.

Nous supposerons dans notre étude que les équipes de deux ambulanciers sont déjà constituées mais elles seront contraintes par des horaires de travail, ceux actuellement définis par la CA. Le nombre d'équipes travaillant en même temps varie entre 1 et 9 en fonction de la période de la journée. Les équipes sont affectées à un véhicule de départ, mais elles peuvent en changer au cours de la journée. Les deux types de véhicules sont $C$ (classique) et $A$ (médicalisable) répartis sur deux dépôts (Trousseau et Bretonneau).

Les demandes de transports ne sont plus caractérisées par un ensemble d'équipements ou type de véhicule requis mais seulement par un type parmi les trois suivants :

- Le transport classique : les demandes de ce type peuvent être assurées par n'importe quel type de véhicule sans contrainte particulière.
- Le transport contagieux : les demandes de ce type doivent être réalisées uniquement par des ambulances de type $C$. Après le transport du patient contagieux, l'ambulance doit être désinfectée à un dépôt par l'équipe ayant effectué le transport. Cependant, soit l'opération est réalisée tout de suite après le transport, soit l'équipe change de véhicule et revient désinfecter le véhicule plus tard dans la journée.
- Le transport médicalisé : les demandes de ce type doivent être réalisées uniquement par des ambulances de type $A$ et nécessitent un médecin, provenant du service d'origine du patient, pour sa surveillance médicale pendant le transport. Après le

transport, le médecin doit être ramené à son service d'origine. Cependant, ce retour peut être réalisé avec un autre patient à bord si et seulement si le temps de détour total causé par ce patient est inférieur à une certaine durée (temps maximal perdu par le médecin dans le transport détourné). Le temps pris pour déposer le médecin n'est pas négligé et supposé constant.

Enfin, les demandes du SAMU seront modélisées par une demande classique caractérisée par un temps de réponse très court et une forte priorité.

Afin de décrire de manière formelle le problème dynamique, nous allons présenter la notation reprise, et complétée, de l'étude précédente, ainsi que la modélisation d'une solution s'appuyant aussi sur l'étude précédente.

#### 7.2.1.1 Notations

Les notations de l'étude précédente ne permettent pas de définir entièrement le problème considéré dans cette section. D'une part des informations liées à la nature dynamique du problème et des contraintes additionnelles doivent être ajoutées, d'autre part une distinction entre les équipes et les ambulances doit être faite. La nouvelle notation du problème est la suivante :

- Un ensemble $\mathcal{M}$ d'équipes : une équipe, constituée de deux ambulanciers, est affectée à un véhicule de départ. Chaque équipe $k \in \mathcal{M}$ est caractérisée par :
  - une date de disponibilité qui correspond à l'heure d'embauche de l'équipe $k$,
  - un dépôt $D_k$ représentant le lieu de départ de l'équipe et le lieu où doit revenir l'équipe après sa journée de travail,
  - une date à laquelle l'équipe doit revenir au dépôt qui correspond à l'heure de fin de travail. Pour ne pas être trop contraint sur l'heure de fin de travail, les équipes peuvent effectuer des heures supplémentaires. Cependant, les solutions avec des heures supplémentaires sont pénalisées par un coût défini par : $e^{(r_k - T)}$ en notant $r_k$ la quantité de travail supplémentaire effectuée par l'équipe $k$ et $T$ une durée de travail supplémentaire considérée comme acceptable.
- Un ensemble $\mathcal{V}$ de véhicules. Les équipes peuvent changer plusieurs fois de véhicules au cours de la journée. Cette opération ne peut s'effectuer qu'à l'un des deux dépôts et nécessite un temps de changement noté $p_{ch}$. Chaque véhicule $v \in \mathcal{V}$ est caractérisé par :
  - un type : $C$ pour les ambulances classiques, $A$ pour les ambulances médicalisables. Chaque type de véhicule existe en nombre limité.
  - un coût $w^v_{i,j}$ si le véhicule $v$ réalise la demande $i$ puis la demande $j$ (déplacement vers le point de départ de cette demande). Ce terme dépend encore des frais kilométriques et du type du véhicule (le type $A$ coûte plus cher que le type $C$).
  - un dépôt $D_v$ qui représente l'emplacement du véhicule à chaque début et fin de journée.
- Un ensemble $\mathcal{R}$ de demandes : chaque demande $i \in \mathcal{R}$ est caractérisée par :
  - un point de départ $dep_i$ et d'arrivée $dest_i$ correspondant respectivement à l'unité de soin de départ et d'arrivée du patient,
  - un type de transport parmi {classique, contagieux, médicalisé},

  - une priorité, notée $Prio_i$ comprise entre 0 et 4 : la priorité 0 correspond à la demande de transport la plus urgente (en particulier celle émanant du SAMU),
  - un temps $t_{dep_i,dest_i}$ de déplacement du véhicule entre le lieu de départ $dep_i$ et de destination $dest_i$ de la demande $i$,
  - une durée $p_i$ de prise en charge du patient par les ambulanciers pour effectuer le transport en dehors du véhicule,
  - une date à partir de laquelle le transport peut commencer, notée $e_i$. Nous associons aussi une fenêtre de temps à l'intérieur de laquelle le transport doit commencer. Cette fenêtre de temps dépend de la priorité de la demande. Lorsque la demande est urgente, la fenêtre de temps associée est étroite, tandis que si la demande est moins urgente, la fenêtre de temps est plus large. Pour prendre en compte cette notion, une demande $i$ est associée à une fenêtre de temps définie par $[e_i, e_i + \alpha_i]$ où $\alpha_i = \tau_0 + \delta \times Prio_i$ avec $\tau_0$ la valeur du retard maximum autorisé pour la plus urgente des demandes et $\delta$ la différence en minutes entre deux niveaux de priorités.
  - un coût de sous-traitance noté $cs_i$. Ce coût dépend de la demande mais pas de la compagnie privée appelée pour l'assurer. Ce coût est strictement plus grand que le coût de l'affecter à une ambulance du CHRU. Par conséquent, l'objectif pour le CHRU est d'assurer le plus de transports possible. Nous appellerons "demande non satisfaite" une demande de transport qui est affectée à une compagnie privée, ou encore une demande non assurée par les ambulances du CHRU.
- Le temps de désinfection est noté $p_{des}$.
- Le temps pris pour déposer le médecin est noté $p_{med}$.
- Le temps maximal de détour total accepté par un médecin est noté $t_{max}$.

Les données liées à la nature dynamique du problème sont les suivantes :
- l'état de chaque véhicule à un instant $t$ ("utilisé par une équipe", "doit être désinfecté", "disponible"), et leurs positions.
- La date de disponibilité de chaque équipe qui correspond soit à l'heure d'embauche, soit à la fin de leur activité courante (transport en cours, désinfection, etc.), soit la date courante s'il n'exécute aucune activité.
- La localisation ou le point de départ de chaque équipe à sa date de disponibilité.

Comme l'étude précédente, le problème consiste à affecter les demandes de transports aux équipes du CHRU, mais cette fois-ci en temps réel et de déterminer les routes de chaque véhicule. La fonction objectif est de minimiser les coûts de transports assurés par le CHRU, plus les coûts des transports sous-traités, plus une pénalité pour les heures supplémentaires réalisées par chaque équipe.

#### 7.2.1.2 Modélisation

La modélisation est basée sur celle présentée dans l'étude du problème statique. Une solution à un instant $t$ est constituée d'un ensemble de routes, avec une route par équipe. Une route est une liste de sommets, définie de la manière suivante :
- un sommet '*' symbolisant la position de l'équipe à la fin de son activité courante ou sa position courante si elle n'effectue aucune activité,

- des sommets 'activités' représentant :
  - soit une activité de transport d'un patient,
  - soit une activité de changement de véhicule à un dépôt,
  - soit une activité de désinfection d'un véhicule à un dépôt, ces sommets sont insérés après les transports contagieux d'un patient,
  - soit une activité de retour de médecin à son unité de soin d'origine, ces sommets sont insérés après les transports médicalisés,
- un sommet $D_k$ représentant le dépôt de l'équipe $k$.

Par rapport à la modélisation de l'étude précédente, la notion de longueur $d_{i,j}$ sur un arc entre deux sommets $i$ et $j$ est étendue. La longueur représente encore le durée nécessaire entre le début de l'activité $i$ et le début de l'activité $j$. Mais le calcul de $d_{i,j}$ dépend donc des sommets $i$ et $j$. Les principaux cas sont les suivants :

- si $i$ et $j$ représentent des demandes de transports : $d_{i,j} = t_{dep_i,dest_i} + p_i + t_{dest_i,dep_j}$ (identique à la modélisation précédente, section 7.1.1.2),
- si $i$ est une demande de transport et $j$ un dépôt $D$ (cas d'une désinfection d'un véhicule, d'un changement de véhicule ou encore d'une fin de travail d'une équipe) : $d_{i,j} = t_{dep_i,dest_i} + p_i + t_{dest_i,D}$
- si $i$ est un dépôt $D$ et $j$ une demande de transport : $d_{i,j} = p + t_{D,dep_j}$ avec $p$ égal à $p_{ch}$ pour un changement de véhicule ou égal à $p_{des}$ pour une opération de désinfection.
- si $i$ est une demande de transport et $j$ le retour de médecin au point $dep_u$ : $d_{i,j} = t_{dep_i,dest_i} + p_i + t_{dest_i,dep_u}$ (dans le cas d'un retour direct du médecin après le transport du patient : $dep_u = dep_i$).

**Exemple :**
La figure 7.3 montre un exemple avec 5 demandes de transports, 2 équipes, 4 services de soins et un seul dépôt. Les demandes sont $\mathcal{R} = \{a, b, c, d, e\}$ avec $a$, $b$, et $c$ des demandes de transports classiques, $d$ une demande de transport médicalisée et $e$ une demande de transport contagieux.

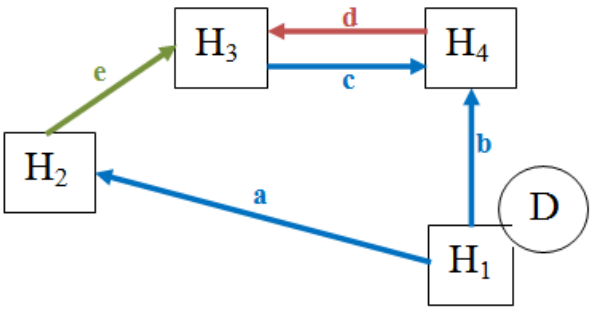


FIG. 7.3 – Illustration de l'exemple

Les figures 7.4 et 7.5 montrent une solution pour cet exemple. Le sommet '*' correspond à la position courante des équipes à l'instant $t$. Dans cet exemple, l'équipe 1 réalise le transport $e$ puis change son véhicule de type $C$ pour un autre véhicule, assure la de-

mande $a$ et retourne au dépôt pour désinfecter le véhicule qu'elle a utilisé pour le transport contagieux. L'équipe 2 commence par changer son véhicule de type $C$ pour un véhicule de type $A$, réalise le transport $b$ puis le transport médicalisée $d$, ramène le médecin (sommet $d'$ qui a été inséré après le transport $d$), mais le retour s'effectue en même temps qu'un autre transport de patient $c$. Enfin l'équipe retourne au dépôt pour récupérer son véhicule d'origine.

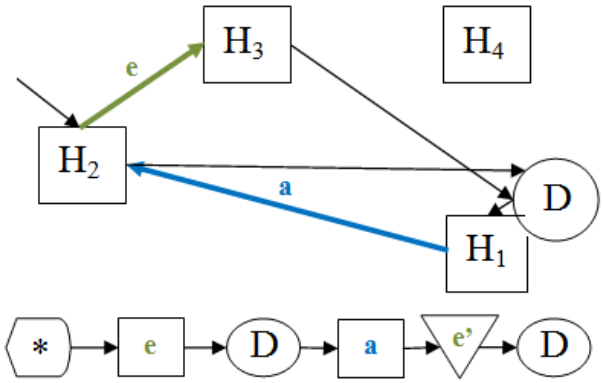


FIG. 7.4 – Route de l'équipe 1

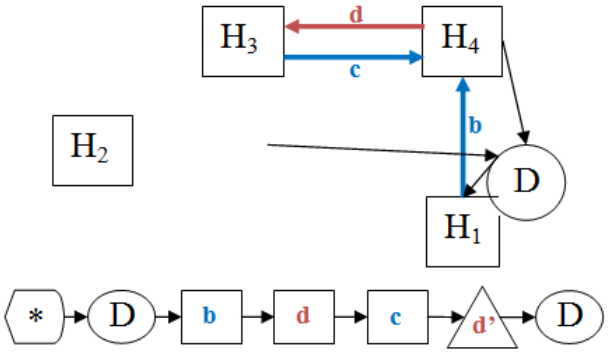


FIG. 7.5 – Route de l'équipe 2

### 7.2.2 Recherche tabou avec mémoire adaptative dans le cas dynamique

La méthode de résolution dans le cas dynamique est basée sur la recherche tabou avec mémoire adaptative du cas statique. Cette méthode de résolution a déjà fait ses preuves en termes de performance sur d'autres problèmes dynamiques de tournées de véhicules. Initialement décrite dans l'article de Taillard et al. [185] dans le cas statique, la méthode fut rapidement reprise pour résoudre ce même problème dans sa version statique (Gendreau et al. [82]) avec une implémentation parallélisée se basant sur l'article de Badeau et al. [9]. Peu de temps après, la méthode fut améliorée en intégrant la notion de détournement de véhicule (Ichoua et al., 2000 [102]). Cette notion consiste à détourner un véhicule de sa destination prévue pour desservir un nouveau client proche de sa localisation courante, contrairement aux cas précédents qui supposent la prochaine destination d'un véhicule

fixée. Enfin, une autre extension de cette méthode est présentée dans l'article de Gendreau et al. en 2006 [81]. Dans cette extension, les clients représentent des demandes de transports de biens, avec pour chaque demande un point de départ et un point de destination. Le problème abordé par les auteurs est une application typique des problèmes des services de courriers dans un environnement dynamique.

Pour adapter la méthode dans le cas dynamique avec l'ensemble des nouvelles contraintes, nous nous sommes inspirés de ces articles. Nous avons aussi intégré la notion de détournement de véhicule mais sans parallèlisation de la méthode. La recherche tabou avec mémoire adaptative étant déjà décrite dans la section 7.1.3, cette section présente uniquement les modifications de la méthode pour cette adaptation.

#### 7.2.2.1 Structure générale

L'idée de l'algorithme est sensiblement la même que dans le cas statique. L'algorithme 7 $TSAM_{Dyn}$ décrit les grandes étapes de la méthode. Cependant, la résolution est effectuée à chaque nouvelle demande de transport apparue à l'instant $t$ avec la mémoire adaptative (AM) de la précédente résolution à l'instant $t - \Delta$. Lorsqu'une nouvelle demande apparaît, la liste des routes de la mémoire doit être mise à jour : les demandes de transports réalisées sont supprimées, les positions des équipes et leurs dates de disponibilité sont modifiées. La nouvelle demande est considérée à cette étape comme non satisfaite. Contrairement au cas statique, lors de la reconstruction d'une solution à partir des routes de la mémoire adaptative, certaines demandes peuvent être non satisfaites, c'est-à-dire qu'elles ne sont assurées par aucune des ambulances du CHRU et sont donc par défaut sous-traitées. Des étapes consistant à essayer de réinsérer les demandes non satisfaites dans les routes des véhicules ont donc été ajoutées dans l'algorithme. Le principe de décomposer et recomposer une solution durant I itérations reste le même. Cependant les deux sous-ensembles des routes de la solution ne sont plus nécessairement disjoints. De plus, les deux exécutions successives de la recherche tabou sur les sous-ensembles des routes utilisent la même liste tabou. Enfin, une étape d'amélioration de la solution courante à la ligne 11 de l'algorithme a été ajoutée de manière à améliorer les affectations des dépôts de changement des ambulances.

#### 7.2.2.2 La mémoire adaptative

Une action supplémentaire vient s'ajouter à la mémoire adaptative, elle consiste à mettre à jour les routes suite à une nouvelle demande à un instant $t$ : supprimer les demandes réalisées et modifier les positions des équipes et leurs dates de disponibilité. Pour cette dernière modification, si une équipe est en train d'effectuer un transport (soit d'un patient soit d'un médecin) sa date de disponibilité est égale à la date de fin du transport, et sa position finale devient le prochain point de départ de l'équipe. Si une équipe n'effectue aucun transport, sa date de disponibilité est égale à l'instant $t$ et le point de départ de l'équipe est sa position courante. Une équipe peut être donc en chemin vers la demande de transport suivante. Cette gestion de mise à jour de la mémoire adaptive permet ainsi la notion de détournement d'une ambulance si une demande de transport apparaît dans

**Algorithme 7** $TSAM_{Dyn}$

1: $AM\,(t) \leftarrow$ Mise à jour de $AM\,(t-\Delta)$
2: **Pour** $j = 1$ to $J$ **faire**
3: $S \leftarrow$ solution construite à partir de $AM\,(t)$ (les demandes n'apparaissant pas dans $S$ constituent les demandes non satisfaites)
4: Insertion autant que possible des demandes non satisfaites dans les routes de $S$
5: **Pour** $i = 1$ to $I$ **faire**
6: *Décomposition* : création de deux sous-ensembles de routes $S_1$ et $S_2$ à partir de $S$
7: Appliquer la recherche tabou sur $S_1$ pour obtenir $S'_1$
8: Appliquer la recherche tabou sur $S_2$ pour obtenir $S'_2$, en gardant la même liste tabou
9: *Reconstruction* : rassembler $S'_1$ et $S'_2$ pour créer une nouvelle solution $S'$
10: $S \leftarrow S'$
11: Amélioration de $S$ (réorganisation des dépôts de changement de véhicules)
12: Insertion autant que possible des demandes non satisfaites dans les routes de $S$
13: **Fin pour**
14: Ajouter les routes de $S$ dans $AM\,(t)$
15: **Fin pour**
16: Post-optimisation sur la meilleure solution trouvée et ajout des routes dans $AM\,(t)$

le voisinage de sa position actuelle. Pour toute équipe dont l'horaire de fin de travail est inférieure ou égale à $t$, on supprime les routes qui lui sont affectées.

L'action d'ajout de routes dans la mémoire adaptative ne diffère pas du cas statique. Cependant, pour la construction d'une nouvelle solution (ligne 3 de l'algorithme 7 $TSAM_{Dyn}$) il faut tenir compte d'un nouveau facteur. Les ambulanciers peuvent changer de véhicules, une route de la mémoire adaptative requiert donc la disponibilité simultanée d'une équipe d'ambulanciers donnée et d'une ambulance donnée. Après la sélection d'une route de la mémoire adaptative pour construire une solution, il faut interdire de sélectionner, pour compléter la solution, les routes de la mémoire adaptative utilisant au même moment cette ambulance ou cette équipe.

#### 7.2.2.3 Procédure d'initialisation

Puisqu'au début de la journée ($t = 0$), la Centrale des Ambulanciers a connaissance d'approximativement 30% des demandes de transports qui devront être effectuées dans la journée. La mémoire adaptative est initialisée avec plusieurs solutions construites à partir de ces demandes. Pour construire une solution, les véhicules sont tout d'abord placés de manière aléatoire dans une liste. Puis les demandes sont traitées dans un ordre arbitraire : une demande est assurée par le premier véhicule de la liste qui peut la satisfaire en respectant toutes les contraintes. Lorsqu'une demande de transport médicalisé ou contagieux est insérée dans une route, les sommets représentant les changements de véhicules ou les opérations de désinfection ou encore les retours des médecins sont également ajoutés afin que cette route demeure réaliste. Comme pour l'initialisation du cas statique, la recherche

tabou avec un petit nombre d'itérations est exécutée sur chaque solution initiale.

#### 7.2.2.4 Procédure d'amélioration

Pour améliorer une solution, deux types de procédures ont été ajoutées : "Insertion des demandes non satisfaites" (lignes 4 et 12) et "réorganisation des dépôts de changements des véhicules" (ligne 11).

Après la construction d'une solution à partir de la mémoire adaptative, certaines demandes de transports peuvent être non satisfaites. Par défaut, ces demandes sont sous-traitées aux ambulances du privé et génèrent un coût. Pour améliorer la solution, les demandes doivent être insérées dans la solution actuelle. La procédure consiste donc à essayer d'insérer les demandes une par une dans l'une des routes de la solution à condition que cette insertion réduise le coût de la solution. Quand un transport médicalisé est inséré, la procédure insère aussi les sommets correspondant au changement de véhicule et le sommet correspondant au retour du médecin. Le traitement est semblable pour les demandes de transports contagieux avec les sommets de changement de véhicule et le sommet de l'opération de désinfection.

La deuxième procédure d'amélioration est due au fait que certaines demandes de transports ont besoin d'un changement de véhicule (de type C ou A) obligeant les équipes à emprunter des ambulances plusieurs fois par jour et à différents dépôts. Les routes des équipes évoluant dynamiquement, il est parfois intéressant de modifier le dépôt où le changement de véhicule s'effectuera. Cette procédure consiste à réaffecter les dépôts de changement des véhicules dans l'ordre croissant des dates de changement des équipes. Le dépôt le plus prêt possible de l'équipe lui est affecté sachant qu'une ambulance ne peut être empruntée que par une seule équipe à la fois.

#### 7.2.2.5 Diversification

La procédure de diversification est identique à celle de l'algorithme du cas statique. Lorsque la meilleure solution a été générée plus de trois fois durant les $I$ itérations, l'ordre de sélection des routes dans la mémoire adaptative est inversé en commençant par les routes de moins bonne qualité.

#### 7.2.2.6 Décomposition et reconstruction des solutions

La phase de décomposition des routes d'une solution permet de diminuer le temps de résolution car elle réduit l'ensemble des solutions du voisinage de la recherche tabou. Contrairement au cas statique, la résolution ne considère pas toutes les demandes de transports. Certaines demandes ont déjà été traitées et d'autres ne sont pas encore connues. Les instances à résoudre à un instant $t$ sont en conséquence moins importantes. L'une des améliorations de l'algorithme au cas dynamique a été de ré-augmenter l'ensemble des solutions du voisinage de la recherche tabou pour pouvoir continuer à améliorer les solutions tout en gardant un temps de résolution raisonnable. Pour cette raison, l'ensemble des routes $S_1$ et

$S_2$ ne sont plus deux sous-ensembles disjoints. A chaque itération $i$, les routes correspondant à la $(i\ mod(m))^{\text{ième}}$ équipe est sélectionnée avec les mêmes approches : géographique ou temporelle. Cependant, les $m \times 2/3$ routes les plus similaires selon l'approche choisie sont rassemblées pour former l'ensemble $S_1$, et les $m \times 2/3$ les moins similaires forment l'ensemble $S_2$. Après la décomposition, la recherche tabou est exécutée sur chaque sous-ensemble de routes successivement mais en utilisant la même liste tabou contrairement au cas statique. La reconstruction de la solution courante consiste simplement à réaliser l'union des routes de $S_2'$ et celles de $S_1'$ (qui ne sont pas dans $S_2'$).

#### 7.2.2.7 Post optimisation

Tout comme le cas statique, une post optimisation est appliquée sur la meilleure solution trouvée à la fin de l'algorithme. Cette opération utilise le même processus qu'à l'initialisation : appliquer la recherche tabou avec un faible nombre d'itérations mais sur toutes les routes de la solution simultanément.

#### 7.2.2.8 Recherche tabou

La recherche tabou dans le cas dynamique présente la même structure générale que la recherche tabou dans le cas statique. La liste tabou et le critère d'arrêt sont également identiques. L'ensemble des solutions du voisinage est déterminé encore par l'opérateur CROSS exchange. Cependant, il a fallu l'adapter pour prendre en compte les contraintes additionnelles, et en particulier les différents types de sommets des routes. Un certain nombre d'échanges de sous-segments sont interdits de manière à toujours obtenir une solution réalisable. Par exemple, lorsqu'un sous-segment d'une route contenant une demande de transport médicalisé est échangé avec un autre sous-segment d'une autre route, il faut vérifier :

- soit que le premier sous-segment est échangé avec les sommets représentant le changement d'ambulance pour un type A,
- soit que le premier sous-segment est inséré dans la deuxième route à un point où l'équipe possède une ambulance de type A.

De plus, certains sommets ne peuvent être affectés qu'à une route bien précise. C'est le cas par exemple du sommet symbolisant la désinfection. Si un transport contagieux a été effectué par une équipe, mais que l'ambulance n'est pas encore désinfectée, l'équipe peut tout de même emprunter une autre ambulance pour continuer à assurer d'autres transports. Cependant, la résolution doit prendre en compte le fait que l'équipe doit revenir désinfecter l'ambulance. Le sommet symbolisant cette action ne doit donc pas être affecté à une autre route.

Contrairement au cas statique, le calcul d'une solution voisine de la solution courante ne peut pas être effectué avant de réaliser l'échange des sous-segments car la fonction objectif prend en compte le retard des équipes. Lorsque deux sous-segments de deux routes différentes sont échangés, la solution obtenue n'est pas nécessairement réalisable. En particulier, il faut vérifier que toutes les demandes sont traitées à l'heure (fenêtre de temps)

et que la date de retour du médecin à son service d'origine est valide. Enfin, il faut tenir à jour un planning d'utilisation des ambulances pour calculer l'éventuel temps d'attendre d'une équipe devant changer de véhicule. Un temps d'attente trop long peut rendre la route irréalisable.

### 7.2.3 Résultats expérimentaux

Le système d'information mis en place à la centrale pour aider les régulateurs à planifier les demandes de transports (PTAH) permet d'obtenir quelques renseignements sur les demandes réalisées chaque jour. Malheureusement, toutes les données sur ces demandes de transports ne sont pas systématiquement rentrées dans le système. Si un service n'effectue pas de demandes par saisie sur le logiciel PTAH, mais directement par téléphone à la Centrale, les régulateurs ne sont pas obligés de saisir la demande et peuvent la traiter directement. Nous n'avons donc obtenu de l'hôpital qu'un ensemble de données incomplet. Cependant, nous avons extrait des caractéristiques, de manière à générer des instances les plus réalistes possible. Avant de présenter la génération d'instances, nous avons étudié le degré de dynamisme du problème selon les formulations proposées dans la littérature (Larsen [132] et Ghiani et al. [86]). Parmi les données de l'hôpital, nous avons extrait 10 jours sélectionnés aléatoirement avec assez d'informations pour calculer deux degrés de dynamisme $\Psi$ et $\Psi'$ définis par :

$$\Psi = \frac{|\mathcal{R}_d|}{|\mathcal{R}|} \qquad \Psi' = \frac{\sum_{i \in \mathcal{R}} \left[T - (e_i + \alpha_i - t_i^{app})\right] / T}{|\mathcal{R}|}$$

Avec $\mathcal{R}_d$ l'ensemble des demandes dynamiques (qui ne sont pas connues à l'avance), $[0, T]$ l'horizon de la planification et $t_i^{app}$ la date d'apparition de la demande $i$ (les demandes statiques sont caractérisées par $t_i^{app} = 0$). $\Psi$ et $\Psi'$ peuvent varier entre 0 (problème statique) et 1 (problème totalement dynamique). La première définition du degré de dynamisme ne prend en compte ni les dates d'apparition ni les fenêtres de temps. C'est pourquoi Larsen a étendu cette définition et a introduit une seconde définition de degré de dynamisme. Cette seconde définition s'appuie sur la durée durant laquelle les régulateurs peuvent agir pour planifier une demande $i$ : $(e_i + \alpha_i - t_i^{app})$. Le tableau 7.5 montre les degrés de dynamisme observé sur ces 10 jours.

| $Jours$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| $\|\mathcal{R}\|$ | 132 | 91 | 90 | 92 | 115 | 113 | 128 | 99 | 126 | 125 |
| $\Psi$ | 0,62 | 0,62 | 0,66 | 0,66 | 0,53 | 0,55 | 0,52 | 0,58 | 0,47 | 0,58 |
| $\Psi'$ | 0,73 | 0,74 | 0,75 | 0,76 | 0,68 | 0,69 | 0,70 | 0,69 | 0,62 | 0,72 |

TAB. 7.5 – Degrés de dynamisme d'instances réelles

En considérant les fenêtres de temps et les dates d'apparitions, le problème apparaît comme fortement dynamique, sachant que toutes les demandes n'apparaissent pas dans les données (certaines demandes n'étant pas saisies, elles ne sont pas incluses dans les calculs). Le premier jour du tableau représente la journée la plus complète au niveau des données. Nous avons été en mesure de recréer une instance représentant 90% de cette journée pour tester notre algorithme. Nous présentons ces résultats en fin de section.

#### 7.2.3.1 Génération d'instances

De manière à tester l'algorithme dans un environnement dynamique, nous avons généré aléatoirement de nouvelles instances dont les caractéristiques sont encore une fois inspirées de l'activité réelle de la Centrale des Ambulanciers, en prenant en compte l'arrivée en temps réel des demandes. Le langage utilisé pour l'implémentation est le C++. L'ordinateur utilisé est un Pentium(R) 4 CPU 3.00 GHz, 1.00 Go de RAM. Les instances générées possèdent les caractéristiques suivantes :

- La période est d'une journée, de 6h00 à 21h00.
- Les données associées aux équipes sont les suivantes :
  - Le nombre d'équipes total est égal à $|\mathcal{M}| = 11$. Les horaires de chaque équipe correspondent aux horaires réels des équipes de la Centrale des Ambulanciers donnés dans le tableau 1.1 de la section 1.3.1.1. Le quota d'heure supplémentaire acceptable a été fixée à $T = 30$ min.
  - Les affectations des équipes aux dépôts sont aussi celles du cas réel (5 à Trousseau et 6 à Bretonneau).

- Les données associées aux véhicules sont les suivantes :
  - il y a un total de $|\mathcal{V}| = 15$ véhicules qui se répartissent en 13 de type C et 2 de type A.
  - Les coûts $w^v_{i,j}$ sont calculés en fonction des frais kilométriques, établis à $0,2 \times (t_{dep_i,dest_i} + t_{dest_i,dep_j})$ plus un coût additionnel de 15 si un véhicule de type A est utilisé.
  - 7 ambulances de type C et 1 de type A sont affectées à Trousseau, le reste est affecté à Bretonneau.

- Les données associées aux demandes sont les suivantes :
  - Le nombre total de demandes par jour est invariablement de 130 :
    - le pourcentage de demandes connues dès le début de la journée est généré uniformément entre 25% et 35%.
    - le nombre de transports médicalisés est généré uniformément entre 5 et 20.
    - le nombre de transports contagieux est généré uniformément entre 2 et 8.
    - les transports restants sont des transports classiques. Parmi ceux ci, tous ne sont pas indépendants, certains sont des transports "aller-retour" d'un patient. Etant considérées comme deux demandes de transports classiques, la seconde demande de transport est à réaliser après la première, et elle n'est connue qu'après l'apparition de la première. Il y a 30 demandes de transports "aller-retour" par jour, ce qui correspond donc à un total de 60 transports.
  - Les données numériques sont les suivantes :

| | | |
|---|---|---|
| $Prio_i \in [0,4]$ | $p_i \in [5,40]$ min. | $\tau_0 = 5$ |
| $e_i \sim \mathcal{N}(330,300)$ (matin) | $e_i \sim \mathcal{N}(600,300)$ (après midi) | $\delta = 5$ |
| $p_{max} = 15$ min. | $p_{ch} = 10$ min. | $p_{med} = 5$ min. |
| $p_{des} = 60$ min. | | |

  - Les demandes qui ne sont pas connues à l'avance arrivent entre 10 et 240 minutes avant la date effective du début au plus tôt du transport.

– Le coût de sous-traitance d'une demande est défini par $1,5 \times t_{dep_i,dest_i} + 60$ (60 représente un forfait fixe de mobilisation) plus une constante additionnelle si une ambulance de type A est utilisée, ou si une désinfection est nécessaire après le transport.

Cinq jeux de données constitués de 200 instances chacuns, ont été générés. Ces jeux de données se différencient par le nombre de demandes émanant du SAMU représentées par des demandes de transports classiques. Ces demandes sont connues entre 5 et 40 minutes en avance seulement. Elles possèdent la plus forte priorité ($Prio_i = 0$). Ces 5 types d'instances (Type I, II, III, IV et V) correspondent respectivement à l'hypothèse que 0%, 5%, 10%, 15% et 20% des 130 demandes proviennent du SAMU. Dans le cas réel, la CA est confrontée en moyenne à un ratio de 10% des demandes. Le tableau 7.6 montre la moyenne des deux degrés de dynamismes pour les différents jeux de données.

| Type d'instances | I | II | III | IV | V |
|---|---|---|---|---|---|
| $\Psi$ | 0,73 | 0,74 | 0,75 | 0,75 | 0,76 |
| $\Psi'$ | 0,77 | 0,78 | 0,79 | 0,80 | 0,81 |

TAB. 7.6 – Degrés de dynamisme des instances générées

Les degrés de dynamisme des instances générées sont plus importants que les données réelles car nous avons considérés que chaque demande non saisie dans le système était une demande non connue à l'avance.

#### 7.2.3.2 Paramètres de l'algorithme

Après des tests préliminaires, les paramètres de l'algorithme ont été ajustés aux valeurs suivantes :

| | |
|---|---|
| Nombre de solutions initiales | $Nb_{init} = 50$ |
| Taille de la mémoire adaptative ($50 \times \lvert\mathcal{M}\rvert$) | $= 550$ routes |
| Nombre d'itérations ($I = 2 \times \lvert\mathcal{M}\rvert$) | $J = 5$ et $I = 22$ |
| Paramètre d'itérations de la tabou | $A = 7$ et $B = 3$ |
| Probabilité de sélection $\rho$ | $= 1 - 1/4$ |
| Nombre d'itérations pour la tabou à l'étape d'initialisation | $= 5$ |
| Nombre d'itérations pour la tabou à l'étape de post optimisation | $= 10$ |

Le délai de réponse de la méthode est un point important, elle est exécutée à l'arrivée de chaque nouvelle demande, et doit proposer rapidement une solution au régulateur. Or ce délai dépend essentiellement du nombre d'itérations $J$ de la boucle principale (ligne 2 de l'algorithme 7). Nous avons donc mené une étude sur la valeur de $J$ afin de dégager un compromis entre une bonne qualité de solution et un temps de réponse raisonnable. Chaque itération $j$ de cette boucle construit une meilleure solution locale $So_j$ qui sert à faire évoluer la mémoire adaptative. Au sortir de la boucle, la meilleure des solutions trouvées (probablement $So_J$) est proposée au régulateur après une étape de post optimisation. Après avoir fixé arbitrairement $J = 20$, nous avons observé l'évolution des solutions $So_j$,

plus précisément l'amélioration relative $\Delta_j$ de $So_j$ par rapport à $So_1$, la solution trouvée à la fin de la première itération de la boucle.

$$\Delta_j = \frac{f(So_1) - f(So_j)}{f(So_1)}$$

Cette étude a été menée sur des instances de type III telles qu'elles sont décrites dans la section 7.2.3.1, le graphique 7.6 représente l'évolution de la valeur moyenne de $\Delta_j$ sur les diverses instances. Il apparaît qu'au-delà de cinq itérations, l'amélioration est minime, nous avons donc décidé de poser $J = 5$ autrement dit de limiter à 5 le nombre d'itérations de la boucle principale de $TSAM_{Dyn}$. De plus, cela correspond à un temps d'exécution de la méthode d'approximativement 2 secondes.

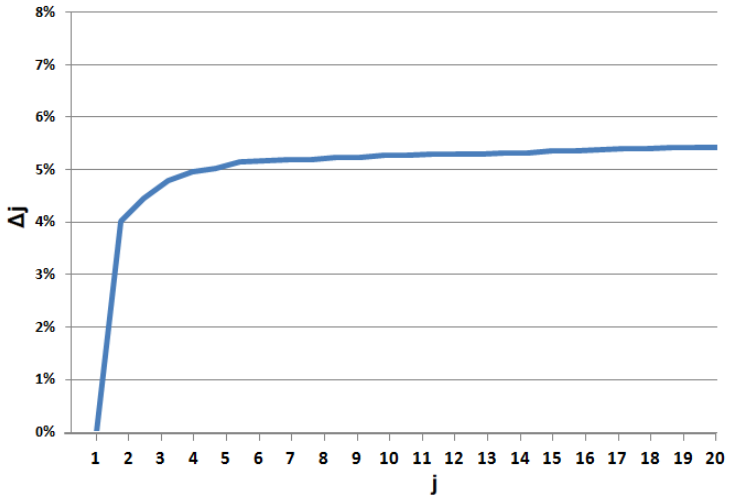


FIG. 7.6 – Evolution de la qualité des solutions en fonction du nombre d'itération $j$

#### 7.2.3.3 Résultats expérimentaux

Tout d'abord, nous avons comparé les solutions données par l'algorithme dans le cas dynamique aux solutions données par une version de cet algorithme dans le cas statique où toutes les données sont connues à l'instant $t = 0$. Dans ce dernier cas, toutes les demandes doivent être insérées dans les solutions initiales. Nous notons $TSAM_{Static}$ l'algorithme dans le cas statique. Ce dernier ne possède pas les mêmes paramètres que l'algorithme du cas dynamique. Contrairement à ce cas, l'algorithme $TSAM_{Static}$ prend en compte l'ensemble des demandes pour une résolution en une seule fois. Ses paramètres ont donc été modifiés pour augmenter son efficacité à obtenir les meilleures solutions possibles. Les nouveaux paramètres sont les suivants : taille de la mémoire adaptative = 1100 routes, $J = 50$, $I = 44$, $A = 25$ and $B = 1$.

Le tableau 7.7 présente les résultats de $TSAM_{Static}$ et les résultats de l'algorithme dans

sa version dynamique noté $TSAM_{Dyn}$. $\Delta$ est la déviation relative définie par :

$$\Delta = \frac{f(Algo_{Dyn}) - f(Algo_{Static})}{f(Algo_{Dyn})}$$

| Types des instances | Moyenne $\Delta$ | Ecart type | Maximum | Minimum | Pourcentage des instances où $f(TSAM_{Dyn}) < f(TSAM_{Static})$ |
|---|---|---|---|---|---|
| Type I | 3,39% | 5,02% | 28,21% | -9,15% | 23,5% |
| Type II | 2,93% | 4,31% | 13,83% | -7,14% | 26% |
| Type III | 2,82% | 4,7% | 22,16% | -21,18% | 26% |
| Type IV | 2,34% | 4,2% | 26,91% | -7,45% | 31% |
| Type V | 2,88% | 4,07% | 14,48% | -5,13% | 27% |

TAB. 7.7 – Comparaison entre $TSAM_{Dyn}$ et $TSAM_{Static}$

Ce tableau montre que quel que soit le nombre de demandes de transports émanant du SAMU, les performances de $TSAM_{Dyn}$ sont stables. La qualité des solutions obtenues est en moyenne 2,34% à 3,39% plus mauvaise que dans les cas statique. La dernière colonne montre qu'entre 23,5% et 31% des instances, $TSAM_{Dyn}$ est meilleur que $TSAM_{Static}$. Pour ces instances, l'exécution de l'algorithme itérativement pour chaque nouvelle demande est meilleure qu'une seule exécution de l'algorithme avec toutes les demandes à la fois.

La structure de l'algorithme est ensuite testée, en particulier l'utilisation de la mémoire adaptative et l'impact de la liste tabou, toujours dans le cas dynamique. Nous notons $TSAM_{Dyn-AM}$ l'algorithme sans la mémoire adaptative (c.-à-d. la mémoire adaptative contient uniquement la meilleure solution trouvée) et $TSAM_{Dyn-LT}$ l'algorithme sans la liste tabou. Les tableaux 7.8 et 7.9 présentent les résultats des comparaison entre $TSAM_{Dyn-AM}$ et $TSAM_{Dyn-LT}$ de la même manière que la comparaison avec l'algorithme dans le cas statique.

| Types des instances | Moyenne $\Delta$ | Ecart type | Maximum | Minimum | Pourcentage des instances où $f(TSAM_{Dyn}) < f(TSAM_{Dyn-AM})$ |
|---|---|---|---|---|---|
| Type I | -6,18% | 5,57% | 3,17% | -48,41% | 92,50% |
| Type II | -4,41% | 5,40% | 11,30% | -23,56% | 89,50% |
| Type III | -4,46% | 4,36% | 4,90% | -30,49% | 88,00% |
| Type IV | -3,83% | 5,31% | 13,46% | -58,78% | 84,50% |
| Type V | -3,74% | 3,60% | 6,51% | -15,19% | 85,00% |

TAB. 7.8 – Comparaison entre $TSAM_{Dyn}$ et $TSAM_{Dyn-AM}$

Les tableaux 7.8 et 7.9 montrent que dans les deux cas, la mémoire adaptative et la liste tabou ont une influence positive sur la qualité des solutions. Pour en moyenne 87,9% et 82,6% des instances, la qualité des solutions décroit sans mémoire adaptative et sans liste tabou respectivement.

| Types des instances | Moyenne $\Delta$ | Ecart type | Maximum | Minimum | Pourcentage des instances où $f(TSAM_{Dyn}) < f(TSAM_{Dyn-LT})$ |
|---|---|---|---|---|---|
| Type I | -4,71% | 5,23% | 4,66% | -46,45% | 83,00% |
| Type II | -3,25% | 5,48% | 14,18% | -22,15% | 86,00% |
| Type III | -3,68% | 3,71% | 4,12% | -20,47% | 84,00% |
| Type IV | -2,87% | 3,33% | 11,59% | -11,05% | 80,00% |
| Type V | -2,68% | 3,23% | 7,27% | -11,91% | 80,00% |

TAB. 7.9 – Comparaison entre $TSAM_{Dyn}$ et $TSAM_{Dyn-LT}$

L'élément le plus important dans la valeur de la fonction objectif de chaque solution est le nombre de demandes sous-traitées au privée. Le tableau 7.10 présente pour chaque algorithme le nombre moyen de demandes sous-traitées. De plus, nous avons étudié une borne inférieure du nombre de ces demandes, notée $BI_{sub}$. Pour déterminer cette borne inférieure, nous avons utilisé le solveur Cplex pour résoudre le PLNE suivant :

- Variables : $\forall i \in \mathcal{R}, \forall k \in \mathcal{M} : x_{ik} = 1$ si l'équipe $k$ réalise la demande $i$, 0 sinon.
- Fonction objectif : Maximiser ( $\sum_{\forall i \in \mathcal{R}} \sum_{\forall k \in \mathcal{M}} x_{ik}$ )
- Contraintes :
  - Une demande $i$ est assurée par au plus une équipe : $\forall i \in \mathcal{R} : \sum_{\forall k \in \mathcal{M}} x_{ik} \leq 1$
  - Si le temps de travail d'une équipe $k$ ne permet pas de réaliser la demande $i$ à l'heure alors : $x_{ik} = 0$
  - Si deux demandes $i$ et $j$ ne peuvent pas être réalisées simultanément à l'heure par une même équipe alors : $\forall k \in \mathcal{M} : x_{ik} + x_{jk} \leq 1$

  Les deux dernières contraintes prennent en compte les fenêtres de temps, les temps de trajets entre sites, les temps de prises en charge du patient hors véhicule et les activités nécessaires pour les transports médicalisés et les transports contagieux. Cependant, nous ne pouvons pas prendre en compte les temps de désinfections et de changement des véhicules étant donné que nous ne savons pas quand ces opérations auront lieu.

| Types | $BI_{sub}$ | $TSAM_{Dyn}$ | $TSAM_{Dyn-AM}$ | $TSAM_{Dyn-LT}$ |
|---|---|---|---|---|
| Type I | 24 | 34,685 | 36,93 | 37,150 |
| Type II | 26 | 37,465 | 39,495 | 39,785 |
| Type III | 27 | 38,555 | 40,225 | 40,855 |
| Type IV | 31 | 41,86 | 42,935 | 43,725 |
| Type V | 32 | 43,78 | 45,005 | 45,495 |

TAB. 7.10 – Nombre moyen de transports sous-traités

La différence entre la borne inférieure et l'algorithme $TSAM_{Dyn}$ est en moyenne de 10 demandes sous-traitées. Cette différence s'explique par les deux dernières contraintes du modèle. Il n'est pas possible de tenir compte de certaines opérations coûteuses en temps (détour pour changer de véhicule, temps de changement de véhicule, temps de désinfecter, etc.). Cependant, notre algorithme trouve en moyenne deux demandes sous-traitées en moins, ce qui n'est pas négligeable pour l'hôpital.

Comme expliqué au début de cette section, une instance réelle a été créée. Elle contient 132 demandes de transport avec 50 demandes connues à l'avance, 6 demandes sont médicalisées, 3 sont contagieuses et 17 émanent du SAMU. Durant cette journée, 36 transports ont été sous-traités en plus des demandes réalisées en retard comparées à la définition des fenêtres de temps du problème. Nous n'avons pas été en mesure de retrouver la valeur de la fonction objectif car il manquait des informations sur les numéros d'équipes des transports assurées. Après une exécution de l'algorithme $TSAM_{Dyn}$, nous trouvons une solution avec 33 demandes sous-traitées. Cette différence de 3 demandes est un gain de l'ordre de 200 Euros pour une journée.

### 7.2.4 Conclusion du modèle dynamique

L'algorithme de la recherche tabou avec mémoire adaptative, modifié pour le cas dynamique en considérant toutes les contraintes du problème, permet d'obtenir des solutions de bonne qualité en un temps très raisonnable ($\approx$ 2 secondes). De plus, cette méthode présente l'avantage d'être à la fois flexible et robuste :

- flexible car elle peut être facilement adaptée pour prendre en compte des pré-affectations afin de tenir compte des choix des régulateurs,
- robuste car si la solution courante devient irréalisable suite à des nouvelles demandes critiques (comme celle du SAMU), une nouvelle solution de bonne qualité peut être rapidement retrouvée grâce aux différentes routes de la mémoire adaptative.

Enfin, parmi les autres avantages de cette méthode dynamique, elle peut être facilement adaptée pour prendre en compte d'autres types d'événements que l'arrivée de nouvelles demandes, comme la panne d'une ambulance ou l'annulation de demandes de transports.

Malheureusement, jusqu'à aujourd'hui, il n'a pas été possible de récupérer des instances réelles complètes pour tester l'algorithme et pouvoir les comparer aux choix des régulateurs (seule une instance a servi de comparaison). De plus, bien que la mise en place à la centrale de cet outil serait une aide à la décision pour les régulateurs, il n'est pas encore possible d'interconnecter l'algorithme à l'outil existant des gestions des demandes de transports : PTAH.

L'une des perspectives de l'étude pourrait être le développement d'une autre heuristique pour le cas dynamique afin d'améliorer l'évaluation de l'algorithme. La génération de colonne pourrait être une méthode très prometteuse pour ce type de problème.

## 7.3 Conclusion du chapitre

Dans ce chapitre, nous nous sommes intéressés au problème d'optimisation des tournées des ambulances auquel sont confrontés les régulateurs de la Centrale des Ambulanciers. L'étude de ce problème s'est décomposée en deux parties. Dans la première partie, le problème est abordé dans sa version statique de la manière la plus générique possible mais

avec néanmoins quelques hypothèse simplificatrices. Dans cette première approche nous avons ramené le problème d'optimisation de tournées d'ambulances à un problème proche du problème du multi-voyageur de commerce. Après une présentation d'une modélisation du problème, deux méthodes de résolution sont proposées. La première est basée sur la Programmation Linéaire en Nombres Entiers (PLNE) en utilisant certaines techniques de pré-processing et des coupes permettant d'améliorer dans certain cas le temps de résolution. La deuxième méthode de résolution est une adaptation d'une heuristique connue dans la littérature : une recherche tabou avec mémoire adaptative [82]. Les résultats expérimentaux entre ces deux méthodes ont montré les performances de l'heuristique et sa capacité à trouver des solutions de bonne qualité dans des temps acceptables.

Dans la seconde partie, le problème est étudié en considérant l'ensemble des contraintes spécifiques à la CA et sa nature dynamique. Contrairement à l'étude précédente, les demandes de transports arrivent en temps réel. Ces demandes doivent donc être affectées aux ambulances du CHRU en fonction de leurs caractéristiques mais aussi de l'état de la flotte de véhicules à l'instant $t$ de la résolution (disponibilités, situations géographiques, etc.). De plus, le nombre d'équipes travaillant en même temps varie entre 1 et 9 en fonction de la période de la journée. Les équipes sont affectées à un véhicule de départ, mais elles peuvent en changer au cours de la journée. Les deux types de véhicules sont $C$ (classique) et $A$ (médicalisable) répartis sur deux dépôts. Les demandes de transports ne sont plus caractérisées par un ensemble d'équipements ou type de véhicule requis mais seulement par un type parmi les trois suivants : classique, contagieux et médicalisé. Suivant le type de transport, des actions, comme ramener le médecin ou encore désinfecter l'ambulance, sont à réaliser après le transport. Enfin, les demandes du SAMU sont modélisées par une demande classique caractérisée par un temps de réponse très court et une forte priorité. Une nouvelle modélisation d'une solution est présentée de manière à prendre en compte toutes ces nouvelles contraintes. L'heuristique mise en place pour résoudre ce problème dynamique, est une adaptation et une amélioration de l'heuristique précédente : la recherche tabou avec mémoire adaptative. Les résultats expérimentaux ont montré qu'une fois encore l'algorithme de la recherche tabou avec mémoire adaptative permet d'obtenir des solutions de bonne qualité en un temps très raisonnable. De plus, cette méthode présente l'avantage d'être à la fois flexible et robuste.