# Mobilité urbaine multimodale

Louise Penz, Eric Sanlaville, Christophe Duhamel

# Nouvelles applications et approches pour la mobilité urbaine et périurbaine

PÔLE INGÉNIEUR
LOGISTIQUE
LE HAVRE NORMANDIE

CAHIERS DE LA LOGISTIQUE

ISSN : 2778-0317

# CAHIERS DE LA LOGISTIQUE

## CL2023 – N° 04

## Nouvelles applications et approches pour la mobilité urbaine et périurbaine

Décembre, 2023

# MOBILITÉ URBAINE MULTIMODALE

**Louise Penz, Éric Sanlaville, Christophe Duhamel**

LITIS, Université Le Havre Normandie
{louise.penz,eric.sanlaville,christophe.duhamel}@univ-lehavre.fr

**RESUME**

Il existe de nombreuses façons de se déplacer dans une ville, en utilisant un véhicule personnel, en utilisant les transports en commun, voire en combinant les deux. Cependant, plus on s'éloigne du centre-ville, plus le choix se réduit au véhicule personnel faute d'accès au réseau de transport en commun. Une alternative consiste à utiliser des véhicules de transport à la demande, plus petits et plus flexibles comme des minibus. On est alors dans le cadre de problème de transport porte-à-porte assurant le transport de passagers de leur origine à leur destination tout en respectant leurs fenêtres temporelles, leur temps de trajet maximum et les capacités des véhicules. Nous étudions une généralisation du problème intégrant le transport en commun existant, comme les trams, et une flotte de navettes flexibles. Deux modèles de Programmation Par Contraintes ont été développés afin de générer des solutions respectant toutes les contraintes. Pour tester les modèles, une instance sur la ville du Havre a été générée, avec les deux lignes de trams existantes ainsi que la ligne future. Les résultats montrent que les modèles sont complémentaires en terme d'objectifs et qu'ils permettent de fournir des solutions que l'on peut ensuite optimiser par d'autres approches.

**MOTS CLÉS:** Transport à la demande, réseau multimodal, programmation par contraintes, solutions initiales.

**ABSTRACT**

There are many ways to travel in a city, using a personal vehicle, using public transportation, or even a combination of the two. However, the farther we go from the city center, the more personal vehicles are mandatory due to lack of access to the public transport network. An alternative is to use smaller, more flexible on-demand transportation vehicles like minibuses. This corresponds to a door-to-door transport problem ensuring the transport of passengers from their origin to their destination while respecting their time windows, their maximum travel times and the capacities of the vehicles. We study here a generalization of the problem integrating existing public transport, such as trams, and a fleet of flexible shuttles. Two Constraint Programming models have been developed in order to generate solutions respecting all constraints. To test the models, an instance of Le Havre city was generated, considering the two existing tram lines as well as the future line. The results show that the models are complementary in terms of objectives and that they allow providing solutions that can then be optimized using other approaches.

**KEYWORDS:** On-demand transportation, multimodal network, contraint programming, initial solutions.

# 1. Introduction

Il existe de nombreux moyens de transport pour se déplacer dans une ville et ses alentours. Ces modes de transport sont plus ou moins flexibles, comme illustré dans la Figure 1. Plusieurs études et recherches ont été menées sur le déplacement, notamment en zone urbaine (Bauchinger et al., 2021). Il ressort que plus les transports sont flexibles, plus ils correspondent à un mode individuel et privé. Aujourd'hui, dans l'union Européenne, deux tiers des voyages se font en voiture. Cependant, l'usage prépondérant des véhicules personnels est la source de multiples problèmes, comme l'augmentation des émissions de gaz à effet de serre ou encore la congestion.

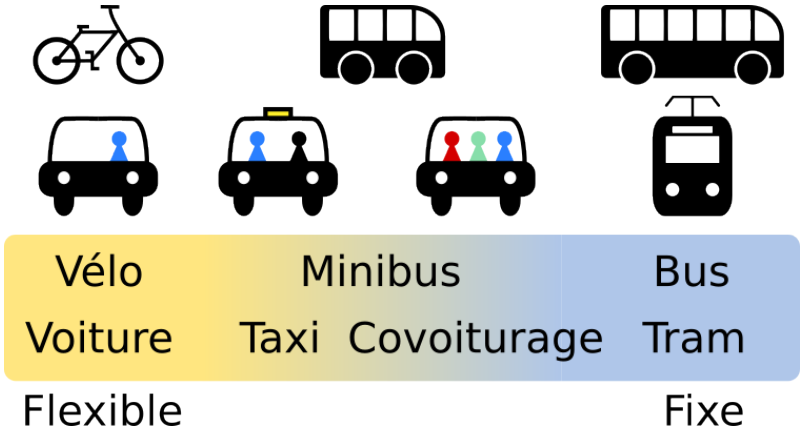



FIGURE 1 – Différents modes de transport dans une ville

Pour pallier ces problèmes, des politiques incitatives sont mises en place pour favoriser les transports en commun, comme le tram, le métro ou les bus. Le basculement massif sur les transports en commun est amené à jouer un rôle important dans la lutte contre le réchauffement climatique et la congestion chronique dans les centres des grandes villes. En outre, les transports en commun contribuent à réduire les inégalités sociales et à amener davantage d'équité. En effet, certaines catégories de personnes n'ont pas accès aux véhicules personnels, que ce soit par manque de moyens financiers, à cause de handicap, ou en raison de contraintes sociétales. En outre, les enfants, les personnes âgées ou certaines personnes handicapées n'ont pas la possibilités de conduire. Pour eux, le transport en commun est donc une nécessité. Dans ce cas, et lorsque l'accès au réseau de transports n'est pas trop compliqué, les lignes permettent de connecter la plupart des points importants dans la ville comme l'hôpital, les services administratifs ou les écoles.

Un autre frein et l'accès inégal au réseau sur le territoire, en zone urbaine comme péri-urbaine ou rurale. En effet, étendre un réseau et mailler les zones peu couvertes induit des coûts d'infrastructure et de fonctionnement inversement proportionnels à la distance au centre ville. De plus, les zones de population en milieu rural sont souvent de faible taille et éparpillées sur le territoire. De ce fait, la rentabilité de lignes de desserte chute rapidement en fonction des distances à parcourir, ce qui conduit mécaniquement à proposer des services à basse fréquence. Les personnes vivant en milieu rural n'ont alors pas d'autre choix que d'utiliser leur véhicule personnel ou de pratiquer le covoiturage, ce qui est source de contraintes et engendre de fortes inégalités sociales. Même en milieu périurbain, voire urbain, les personnes ont tendance à privilégier la voiture à la marche lorsqu'il faut effectuer plus d'un quart d'heure (ou 1 kilomètre) de marche. La Figure 2 montre la carte du Havre avec les lignes de trams existantes ainsi que la future ligne. L'aire en rouge représente les zones qui sont situées à moins d'un quart d'heure de marche de la station de

tram la plus proche et l'aire bleue les zones à moins de 40 minutes de marche. Ainsi, la population vivant en zone bleue a davantage tendance à prendre la voiture. De fait, hors utilisation des parkings relais, ces personnes n'intègrent pas le réseau du tram.

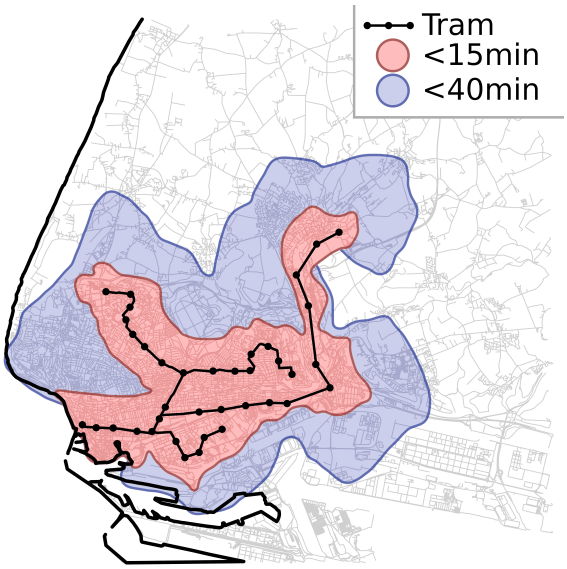



FIGURE 2 – Zones selon la distance à pieds aux trams

Il existe plusieurs moyens pour faciliter le transport en commun. Comme mentionné juste avant, il existe les parking relais localisés à certaines stations ou la solution du covoiturage organisé pour les personnes souhaitant se rendre à la même station. Une autre solution est d'organiser du transport multimodal, où l'on combine une flotte de véhicules plus flexibles, comme les minibus, amenant les personnes aux stations et aux heures désirées. Ceci permet un transport flexible, complémentaire du transport en commun classique, et n'introduisant pas de concurrence entre les deux modes, tout en maintenant le mode flexible loin du centre ville pour éviter la congestion.

## 2. Le problème de transport de passagers à la demande incluant les transports en commun

Le problème du transport de passagers combinant le transport à la demande et les transports en commun est appelé Integrated Dial-A-Ride Problem (IDARP). On définit $R$ l'ensemble des requêtes qui correspondent aux demandes des utilisateurs à satisfaire. Chaque requête $r \in R$ possède une origine $O_r$ et une destination $D_r$. On note $O$ et $D$ l'ensemble de tous les points d'origine et de tous les points de destination respectivement. À chaque point $i \in O \cup D$, on associe une fenêtre de temps $[\underline{f}_i, \overline{f}_i]$. Elle correspond à la période de temps durant laquelle l'utilisateur accepte d'être pris en charge sur ce sommet. Ainsi $\underline{f}_i$ est le premier moment disponible et $\overline{f}_i$ est le dernier moment disponible. Si le véhicule arrive au sommet $i$ avant $\underline{f}_i$, cela engendrera une attente jusqu'au temps $\underline{f}_i$. On définit deux types de requêtes, les requêtes entrantes et les requêtes sortantes. Une requête entrante impose une fenêtre de temps sur son sommet d'origine, c'est-à-dire que le véhicule doit prendre la requête durant sa fenêtre de temps, tandis que la destination n'a pas de fenêtre de temps. Pour les requêtes sortantes, la fenêtre de temps est positionnée sur la destination et non sur l'origine. Pour le confort du passager, chaque requête $r \in R$ définit un temps maximum de trajet $\overline{l}_r$. De plus, chaque requête correspond à un nombre $c_r$ de personnes souhaitant

se déplacer ensemble et impose un temps de service $s_r$ pour entrer ou de sortir du véhicule.

Pour satisfaire ces requêtes, un ensemble $V$ de véhicules est initialement positionné sur un dépôt. Chaque véhicule effectue une tournée de desserte qui commence et termine sur ce dépôt. Les véhicules sont supposés homogènes et possèdent donc une capacité identique $\overline{q}$ en nombre de places. Les utilisateurs peuvent également emprunter le transport en commun. $S$ est l'ensemble des stations utilisables. Les horaires de passages des trams ne sont pas considérés car on suppose dans nos instances que les passages sont fréquents. Le temps d'attente aux stations est une attente moyenne, constante, qui peut être intégrée dans la matrice des durées. Pour un véhicule, le temps de trajet du point $i$ au point $j$ avec $i, j \in (O \cup D \cup S)^2$ est $T(i, j)$ et le temps pour un tram entre $i$ et $j$ avec $i, j \in S^2$ est $T^{pub}(i, j)$.

La Figure 3 montre un exemple de problème, avec 3 requêtes et 2 stations reliées par une ligne. Deux véhicules, notés 0 et 1, sont initialement positionnés au dépôt (carré gris). Le véhicule 0 prend en charge la première moitié de la requête $R_1$ (sommet $P_1$) et la requête $R_0$ en entier (sommets $P_0$ et $D_0$) tandis que le véhicule 1 prend en charge le reste des requêtes. La requête $R_0$ est donc prise par le véhicule 0 pour être amenée à une station. Ensuite la requête prend les transport en commun puis est récupérée à la deuxième station par le véhicule 1 pour terminer son trajet.

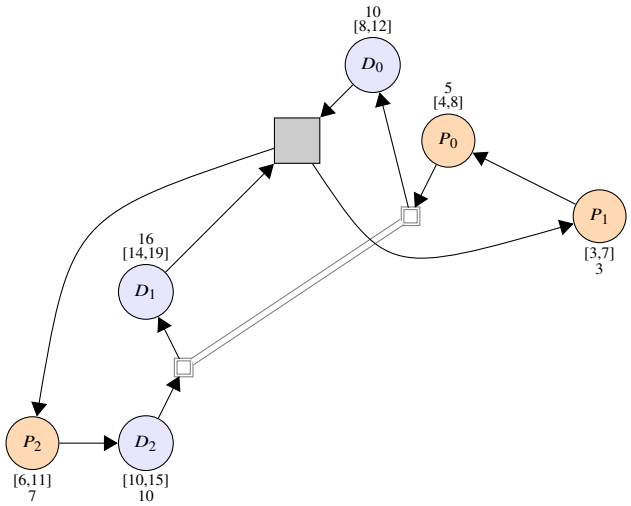



FIGURE 3 – Exemple d'une instance avec sa solution.

Ce problème a déjà fait l'objet de plusieurs études dans la littérature. En 1976, deux rapports, Wilson et al. (1976) et Potter (1976), ont analysé un tel système multimodal sans toutefois traiter l'optimisation du service. Des modèles de Programmation Linéaire en Nombres Entiers (PLNE) ont été développés plus tard, ce qui permet d'obtenir une réponse exacte et optimale. L'objectif à minimiser est souvent le temps de route des véhicules à la demande. Häll et al. (2009) ont proposé une formulation PLNE pour ce problème pouvant résoudre jusqu'à 10 requêtes avec une ligne de transport comprenant 3 stations. Plus récemment, (Posada et al., 2017) ont développé des modèles PLNE pour le IDARP avec des horaires de passage pour le transport en commun, pouvant résoudre des instances jusqu'à 5 requêtes et 4 stations.

Les instances résolues à l'optimum avec ces modèles restent assez modestes. Pour traiter des instances de taille moyenne à grande, il faut utiliser des méthodes heuristiques, qui trouvent des solutions approchées à la solution optimale. Une heuristique d'insertion de requêtes dans une solution partielle a été proposée dans Liaw et al. (1996). Posada and Häll (2020) ont développé la

métaheuristique Large Neigborhood Search (LNS) pour le IDARP. Molenbruch et al. (2021) ont aussi utilisé la LNS, en ajoutant entre autres les horaires de passage des transport en commun. Ces méthodes ont permis de calculer des solutions pour des instances contenant jusqu'à 200 requêtes.

Pour calculer de bonnes solutions sur des grandes instances, il faut nécessairement passer par des heuristiques et surtout par des métaheuristiques. Cependant, pour ces dernières, il est nécessaire de disposer d'une solution initiale, même de mauvaise qualité. Dans la littérature, les solutions initiales ont souvent été calculées en partant de routes vides, puis en insérant les requêtes les unes après les autres. Cette approche permet de trouver une solution initiale dans le cas où insérer des requêtes est assez simple et donc lorsque le problème n'est pas trop contraint. Notre proposition porte ici sur le fait de trouver une solution initiale, même dans les cas contraints. Pour cela, nous avons utilisé des modèles de Programmation Par Contraintes (PPC). Ils permettent de trouver des solutions satisfaisant toutes les contraintes. Par contre, ils ne cherchent pas forcément à optimiser leur qualité.

# 3. La programmation par contraintes

## 3.1 Définition

La Programmation Par Contraintes (PPC) est un domaine de l'Intelligence Artificielle (IA). Elle offre une grande flexibilité en terme de modélisation et ne nécessite pas forcément d'objectif à optimiser. Elle permet donc de tester la faisabilité d'une instance en calculant une solution réalisable. Pour cela, la PPC résout un Problème de Satisfaction de Contraintes, dans lequel elle doit trouver une affectation totale pour les variables telle que les valeurs choisies respectent toutes les contraintes définies.

**Définition 1** *Un Problème de Satisfaction de Contraintes (ou CSP) est défini par un triplet* $\langle \mathcal{X}, \mathcal{D}, C \rangle$ *où :*

- $\mathcal{X} = \{x_1, \ldots, x_n\}$ *est l'ensemble des variables du problème ;*
- $\mathcal{D} = \{D_1, \ldots, D_n\}$ *est l'ensemble des domaines des variables, i.e.* $\forall i \in [\![1, n]\!], x_i \in D_i$ *;*
- $C = \{c_1, c_2, ..., c_m\}$ *est un ensemble de contraintes où chaque contrainte* $c_j$ *est définie par le sous-ensemble du produit cartésien des variables sur lesquelles elle porte :* $c_j(x_{j_1}, \ldots, x_{j_k}) \subseteq D_{j_1} \times \cdots \times D_{j_k}$.

**Définition 2** *Une affectation totale* $\mathcal{A}$ *est définie par* $\mathcal{V}$*, le tuple des valeurs prises par les variables :*

$$\mathcal{V} = \{v_1, \ldots, v_n\} \in \{\mathcal{D}_1, \ldots, \mathcal{D}_n\}$$

Il existe plusieurs types de variables, comme les variables entières dont le domaine est un ensemble d'entiers, les variables booléennes avec comme domaine {Vrai, Faux}. Les modèles présentés dans l'article utiliseront des variables d'intervalle et de séquence d'intervalles. Ces types de variables, présentées dans Laborie and Rogerie (2008), ont été initialement conçus pour les problèmes d'ordonnancement. Un intervalle est une variable de décision caractérisée par quatre attributs : son début, sa fin, sa taille et son état (actif ou non), car un intervalle peut être présent ou non dans la solution.

Le domaine d'un intervalle $i$ est $\{\perp\} \cup \{[debut_i, fin_i) | debut_i, fin_i \in \mathbb{Z}, debut_i \leq fin_i\}$. On associe aussi un domaine pour chacun des attributs de l'intervalle $i$ :

— $debut(i) \in \{\underline{debut}(i), \overline{debut}(i)\}$ ;
— $fin(i) \in \{\underline{fin}(i), \overline{fin}(i)\}$ ;
— $taille(i) \in \{\underline{taille}(i), \overline{taille}(i)\}$ ;
— $presence(i) \in \{Vrai, Faux\}$.

Si l'intervalle $i$ n'est pas présent, l'affectation sera $\perp$. Sinon, les valeurs de $debut_i$ et $fin_i$ seront fixées, avec $taille_i = fin_i - debut_i$.

Une séquence d'intervalles indique une relation entre l'ordre des intervalles et la position relative de leurs dates de départ et d'arrivée. Soit $p$ une séquence d'intervalles sur l'ensemble d'intervalles $I$. Le domaine de $p$ est l'ensemble des permutations possibles de ces intervalles, en prenant en compte le fait que les intervalles peuvent être présents ou non. Une affectation est une permutation des intervalles présents.

La recherche de solutions se fait de plusieurs manières, en propageant les contraintes pour réduire au mieux les domaines, en fixant certaines valeurs pour voir si une solution est faisable, etc. Pour cela, il existe plusieurs solveurs de PPC, qui prennent le triplet variables-domaines-contraintes et renvoient une solution, s'il en existe.

## 3.2 Dans les problèmes de transport de passagers

La PPC a été utilisée par Berbeglia et al. (2011) pour le problème de transport à la demande de passagers. Il correspond à notre problème, sans l'utilisation des transports en commun. Pour traiter leur problème, ils ont utilisé un modèle dit par successeurs, dans lequel chaque sommet (origine et destination) est associé à un entier et à une variable qui indique son successeur direct dans la route. Les successeurs sont donc des variables entières avec comme domaine l'ensemble des sommets possibles pour la suite de la route. D'autres variables sont utilisées, comme le numéro du véhicule à chaque sommet ou le temps d'arrivée sur chaque sommet. Ce modèle a aussi été utilisé par Deleplanque and Quilliot (2013) pour le le problème de transport de passagers avec transferts, dans lequel les utilisateurs sont déposés à certains points et un autre véhicule vient les chercher pour terminer le trajet.

Cappart et al. (2018) travaillent sur le problème du transport de patients. Ce problème est aussi un problème de transport de passagers mais les requêtes sont hétérogènes : une requête peut consister à aller de l'origine à un centre de santé, ou à aller d'un centre de santé à la destination, ou encore d'aller de l'origine à un centre de santé, puis être conduit à la destination (qui peut être le point d'origine). Ce modèle utilise les intervalles comme variables de décision et un intervalle correspond à une étape de la route d'une requête (origine centre de santé ou centre de santé destination). Le début de l'intervalle correspond au moment où le patient entre dans le véhicule et sa fin au moment il en sort.

Liu et al. (2018) travaillent sur le problème du transport de seniors, qui ressemble aux autres problèmes de transport de passagers. Le modèle PPC utilise les intervalles, où chaque intervalle correspond à un sommet. Une requête est donc constituée de deux intervalles, un pour l'origine et l'autre pour la destination. Ham (2023) développe un modèle pour le problème du transport de passagers en utilisant les variables d'intervalles relatives aux requêtes et relatives aux sommets, en souhaitant combinenr les avantages des deux modélisations.

Récemment, Thomas et al. (2020) et Delecluse et al. (2022) ont proposé d'autre types de variables, basés sur les séquences d'évènements, avec une propagation et une affectation des séquences d'intervalles différente. Ils ont testé ces nouveaux types sur le problème du transport de

passagers et ont montré que cela produisait de bons résultats.

Nous avons d'abord testé plusieurs modèles pour le problème du transports de passagers sans transport en commun sur des instances non réalistes afin de valider les modèles précédents. Puis, suite aux résultats, nous avons décidé de nous concentrer sur les modèles avec intervalles, en utilisant les intervalles relatifs aux requêtes et les intervalles relatifs aux sommets afin de les tester sur les instances correspondant à la ville du Havre.

## 4. Le modèle par intervalles de sommets SomMod

Pour le modèle par intervalles de sommets, il y a autant d'intervalles que d'évènements possibles. Un évènement correspond au fait de prendre ou déposer un passager à son origine, à sa destination ou à un arrêt possible de tram. Les arrêts que vont prendre les passagers pour le transports en commun ne sont pas connus à l'avance et le choix de notre solveur est limitant pour la contrainte $noOverlap$. Cette dernière permet d'assurer un temps minimum entre deux intervalles consécutifs dans une séquence, selon une matrice de temps fixe. Il faut donc autant d'intervalles que de couples de sommets, ce qui fait que le modèle va avoir pour intervalles les origines, destinations et tous les couples requête-station pouvant appartenir à une solution. Un exemple est présenté dans la figure 4. L'ensemble des stations avant transport public pour la requête $r$ est déterminé par :

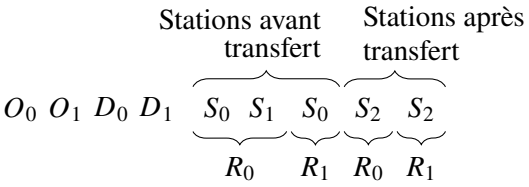
$$S_r^1 = \{S_i \in S \mid \exists S_j \in S :\ T(O_r, S_i) + T^{pub}(S_i, S_j) + T(S_j, D_r) + 3s_r \leq \bar{l}_r\}$$

Les stations après transfert sont calculées de la même manière. On note $\mathcal{S}$ la séquence des origines, destinations, stations avant transfert et stations après transfert.

$R_0$ $S_0 \longrightarrow S_2$
$S_1 \longrightarrow S_2$

$R_1$ $S_0 \longrightarrow S_2$

(a) Les stations que peuvent prendre les requêtes




(b) La liste des intervalles correspondante

FIGURE 4 – Ensemble $\mathcal{S}$ des sommets selon les possibilités des requêtes.

Pour chaque sommet $i \in \mathcal{S}$ on définit un intervalle $x_i$ et pour chaque couple sommet-véhicule $i \in \mathcal{S}$, $v \in V$ on introduit un intervalle $X_{i,v}$. La variable $x_i$ sera utilisée pour les contraintes relatives aux requêtes, comme les fenêtres de temps, et la variable $X_{i,v}$ permet de faire le lien entre les requêtes et les véhicules. Elles seront utilisées dans les contraintes relatives aux véhicules, comme les temps de route. De plus, chaque route du véhicule $v \in V$ est représentée par une séquence $seq_v$. Pour chaque requête $r \in R$, la variable booléenne $u_r$ détermine si la requête utilise les transports en commun. La figure 5 fournit une illustration. On note $S_r^1$ et respectivement $S_r^2$, l'ensemble dans $\mathcal{S}$ des stations avant et après le transfert de la requête $r \in R$. Les intervalles en orange signifient que le passager monte dans un véhicule et les intervalles en bleu qu'il en descend.

Les contraintes du modèle sont :

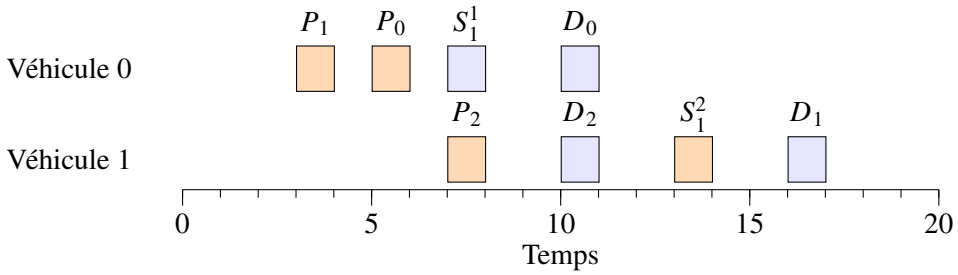



FIGURE 5 – Représentation simplifiée du modèle par intervalles de sommets de la solution de la figure 3.

$$alternative(x_i, [\forall v \in V, X_{i,v}]) \qquad \forall i \in \mathcal{S} \qquad (1)$$

$$noOverlap(seq_v, A) \qquad \forall v \in V \qquad (2)$$

$$\sum_{i \in S_r^1} presence(x_i) = \sum_{i \in S_r^2} presence(x_i) = u_r \qquad \forall r \in R \qquad (3)$$

$$\neg u_r \implies presence(X_{O_r,v}) = presence(X_{D_r,v}) \qquad \forall r \in R,\ \forall v \in V \qquad (4)$$

$$presence(X_{i,v}) \leq presence(X_{O_r,v}) \qquad \forall r \in R,\ \forall i \in S_r^1,\ \forall v \in V \qquad (5)$$

$$presence(X_{i,v}) \leq presence(X_{D_r,v}) \qquad \forall r \in R,\ \forall i \in S_r^2,\ \forall v \in V \qquad (6)$$

$$before(x_{O_r}, x_{D_r}) \qquad \forall r \in R \qquad (7)$$

$$before(x_{O_r}, x_i) \qquad \forall r \in R,\ \forall i \in S_r^1 \qquad (8)$$

$$before(x_i, x_j, T^{pub}(i,j)) \qquad \forall r \in R,\ \forall i \in S_r^1,\ \forall j \in S_r^2 \qquad (9)$$

$$before(x_i, x_{D_r}) \qquad \forall r \in R,\ \forall i \in S_r^2 \qquad (10)$$

$$\underline{f}_i \leq debut(x_i) \leq \overline{f}_i \qquad \forall i \in O \cup D \qquad (11)$$

$$taille(x_i) = s_r \qquad \forall r \in R,\ \forall i \in \{O_r, D_r\} \cup S_r^1 \cup S_r^2 \qquad (12)$$

$$debut(x_{D_r}) - debut(x_{O_r}) \leq \overline{l}_r \qquad \forall r \in R \qquad (13)$$

$$cumulative\Bigg(\sum_{r \in R}\Big(stepAtStart(X_{O_r,v}, c_r) + stepAtStart(X_{D_r,v}, -c_r) + \sum_{i \in S_r^1} stepAtStart(X_{i,v}, -c_r) + \sum_{i \in S_r^2} stepAtStart(X_{i,v}, c_r)\Big)\Bigg) \qquad \forall v \in V \qquad (14)$$

Les sommets ne sont desservis que par un seul véhicule grâce aux contraintes (1). Ces dernières utilisent la contrainte $alternative(i, [i_1, \ldots, i_n])$. Cette contrainte est valide si et seulement si tous les intervalles sont absents ou alors si $i$ et un et un seul intervalle parmi $\{i_1, \ldots, i_n\}$ est présent et que les dates de début et de fin sont égales. Dans notre cas, cela signifie qu'un sommet est pris en charge par un unique véhicule. Les contraintes (2) permettent de prendre en compte les temps de trajet entre les intervalles avec la matrice précalculée $A$ déduite de la structure de $\mathcal{S}$. Les contraintes (3) assurent qu'un seul intervalle de transfert est présent avant et après transfert, dans le cas de transfert. Les intervalles des origines, des destinations, des origines et stations, ou des stations et destinations sont sur la même route grâce aux contraintes (4) à (6). Les intervalles sont dans la beonne séquence chronologique par les contraintes (7) à (10). Dans les contraintes (9), la contrainte $before$ prend un troisième argument qui est le temps minimum

entre les deux intervalles. Cela permet de faire respecter les temps de trajet dans les transports en commun. Les fenêtres de temps aux origines et destinations, les temps de service et les temps maximum de trajet des requêtes sont respectés par les contraintes (11)-(13). La capacité maximale des véhicules est respectée par les contraintes (14). Les fonctions $stepAtStart(x, c)$ sont indexées dans le temps et augmentent la quantité de la valeur $c$ au début de l'intervalle $x$. Ainsi, on augmente de la valeur $c$ sur l'intervalle de l'origine de la requête et on diminue de cette même valeur $c$ à la destination. La fonction *cumulative* permet de cumuler toutes ces fonctions pour toutes les requêtes. Ainsi, à chaque pas de temps, on connaît la quantité de personnes dans un véhicule et on s'assure que cette quantité ne dépasse pas la capacité du véhicule.

## 5. Le modèle par intervalles de requêtes ReqMod

Pour le modèle par intervalles de requêtes, une requête est représentée par deux intervalles. Dans le cas de l'utilisation du transport en commun, le premier intervalle représente la prise en compte du passager par la première voiture. Le début de l'intervalle correspond au moment où le passager entre dans le véhicule et la fin au moment où il en sort. Le second intervalle est la prise en charge du passager par le second véhicule après avoir utilisé les transports en commun, pour aller jusqu'à sa destination. Si le passager ne prend pas de transport en commun, alors seul le premier intervalle sera présent et couvrira le trajet de l'origine à la destination.

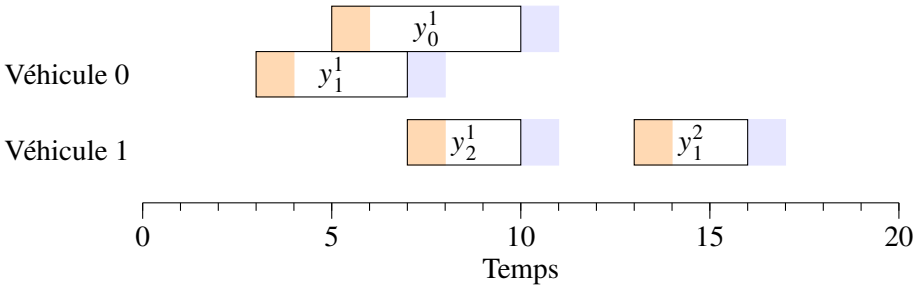



FIGURE 6 – Représentation simplifiée du modèle par intervalles de requêtes de la solution de la figure 3.

On utilise des intervalles par requête et des intervalles par requête-véhicule comme variables de décision. Soit $y_r^1$, respectivement $y_r^2$, l'intervalle de la première étape du trajet ou de la requête complète dans le cas de non transfert, et l'intervalle de la seconde étape. Les intervalles correspondants liés avec un véhicule $v$ sont notés $Y_{r,v}^1$, resp. $Y_{r,v}^2$. La figure 6 montre une représentation pour la solution de la figure 3. Les requêtes $R_0$ et $R_2$ n'utilisent qu'un seul véhicule. Elles sont donc représentées uniquement par l'intervalle $y_r^1$ qui correspond à tout le trajet. La requête $R_1$ effectue un transfert et est donc représentée par l'intervalle $y_r^1$ pour la partie avant le transfert et par l'intervalle $y_r^2$ pour la partie après le transfert. Comme il est possible de ne pas avoir d'intervalle de la seconde partie du trajet, $y_r^2$ est optionnel, tout comme $Y_{r,v}^1$ et $Y_{r,v}^2$. Soit $u_r$ une variable booléenne qui est égale à Vrai si la requête $r$ prend les transports en commun, Faux sinon. Pour chaque requête $r$, la station avant le transfert, resp. après le transfert, est $\phi_r^1$, resp. $\phi_r^2$. Pour ne pas considérer toutes les stations, un prétraitement est réalisé pour chaque requête pour que les domaines de $\phi_r^1$ et $\phi_r^2$ ne contiennent que des stations pouvant potentiellement appartenir à une solution. Les carrés orange représentent le moment ou les passagers montent dans le véhicule et les carrés bleus le moment où ils en descendent. La fin de l'intervalle se situe avant la descente

du passager, mais le véhicule ne peut repartir qu'après ce temps de service.

Les contraintes du modèle sont :

$$alternative(y_r^1, [\forall v \in V, Y_{r,v}^1]) \qquad \forall r \in R \qquad (15)$$

$$alternative(y_r^2, [\forall v \in V, Y_{r,v}^2]) \qquad \forall r \in R \qquad (16)$$

$$debut(y_r^2) - fin(y_r^1) \geq T^{pub}(\phi_r^1, \phi_r^2) + s_r \qquad \forall r \in R \qquad (17)$$

$$\neg u_r \implies taille(y_r^1) \geq T(O_r, D_r) + s_r \qquad \forall r \in R \qquad (18)$$

$$u_r \implies taille(y_r^1) \geq T(O_r, \phi_r^1) + s_r \qquad \forall r \in R \qquad (19)$$

$$taille(y_r^2) \geq T(\phi_r^2, D_r) + s_r \qquad \forall r \in R \qquad (20)$$

$$distMin(i,j) \qquad \forall i \in Y_{r_1,v}^1 \cup Y_{r_1,v}^2, j \in Y_{r_2,v}^1 \cup Y_{r_2,v}^2, \forall r_1, r_2 \in R^2, r_1 \neq r_2, \forall v \in V \qquad (21)$$

$$presence(y_r^2) = u_r \qquad \forall r \in R \qquad (22)$$

$$\underline{f}_{O_r} \leq debut(y_r^1) \leq \overline{f}_{O_r} \qquad \forall r \in R \qquad (23)$$

$$\neg u_r \implies \underline{f}_{D_r} \leq fin(y_r^1) \leq \overline{f}_{D_r} \qquad \forall r \in R \qquad (24)$$

$$u_r \implies \underline{f}_{D_r} \leq fin(y_r^2) \leq \overline{f}_{D_r} \qquad \forall r \in R \qquad (25)$$

$$taille(y_r^1) \leq \overline{l}_r \qquad \forall r \in R \qquad (26)$$

$$u_r \implies fin(y_r^2) - debut(y_r^1) \leq \overline{l}_r \qquad \forall r \in R \qquad (27)$$

$$cumulative\left(\sum_{r \in R} pulse(Y_{r,v}^1 \wedge Y_{r,v}^2, c_r)\right) \leq \overline{q}_v \qquad \forall v \in V \qquad (28)$$

Les requêtes, avant ou après le transport en commun, ne peuvent être transportées que par un seul véhicule par les contraintes (15) et (16). Le temps pour les transports publics est pris en compte par les contraintes (17). Les contraintes (18) à (21) assurent que les temps de trajet sont respectés. Dans le cas où un transfert est effectué, l'intervalle pour la seconde partie du trajet est présent grâce aux contraintes (22). Les intervalles respectent les fenêtres de temps aux origines et destinations par les contraintes (23) à (25) et le temps maximum de trajet par les contraintes (26) et (27). Les contraintes (28) assurent que les véhicules ne dépassent pas leurs charges. La fonction $pulse(y, c)$ augmente de $c$ la quantité au début de l'intervalle $y$ et revient à 0 à la fin de l'intervalle.

# 6. Cas d'étude

## 6.1 Instance du Havre

Pour comparer nos modèles, nous allons étudier un cas construit à partir des donnée de la ville du Havre. Pour cela, 50 requêtes sont générées aléatoirement. Les origines et destinations sont générées dans la zone bleue de la figure 2. Une requête peut être soit entrante, c'est-à-dire qu'il y a une fenêtre de temps uniquement sur son origine, soit sortante avec seulement une fenêtre de temps sur sa destination. Le temps maximal de trajet est limité au double du temps de trajet direct en voiture de l'origine à la destination. Cela permet de favoriser l'utilisation du transport en commun et d'autoriser des détours pour aller chercher ou déposer d'autres requêtes. Les fenêtres de temps ont une largeur de 20 minutes. Une requête a 70% de chance d'être constituée d'une seule personne,

20% de chance de concerner deux personnes et 10% de chance d'impliquer trois personnes. Pour transporter ces personnes, on dispose de 4 véhicules avec une capacité de 6 personnes. L'instance est représentée sur la figure 7, où les points rouges sont les origines et les destinations.

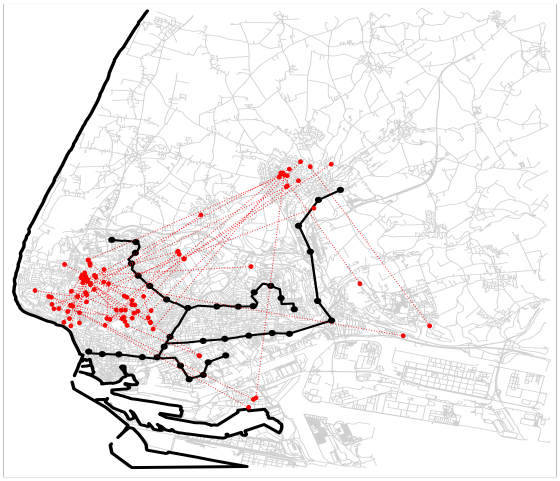


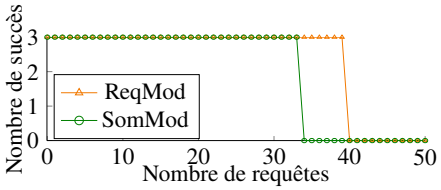
FIGURE 7 – Instance du Havre

## 6.2 Les résultats

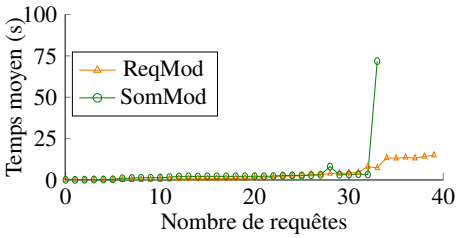
Le solveur utilisé est ILOG CP Optimizer en version 22.1.0. Pour comparer les modèles SomMod et ReqMod, chaque instance est analysée selon le nombre de requêtes prise en compte. Ces dernières sont stockées dans une liste et le solveur est appelé 3 fois sur les $k$ premières requêtes de l'instance avec $k$ variant de 0 à 50. Appeler le solveur 3 fois, chaque fois avec une racine différente du générateur de nombres aléatoires, permet d'obtenir une moyenne sur les résultats. En effet, lors de la résolution, le solveur fait des choix reposant sur l'aléatoire, ce qui peut conduire à des solutions différentes entre deux appels. Nous avons fixé en outre une limite de temps de 300 secondes pour chaque appel du solveur. Ainsi, soit le solveur arrive à calculer une solution avant les 300 secondes et on considère que c'est un succès, soit il ne renvoie pas de réponse et c'est un échec. En effet, l'instance a été construite de telle manière qu'il existe toujours une solution pour toutes les valeur de $k$.

La figure 8a montre le nombre de succès trouvés par chaque modèle. Ce nombre varie entre 0 et 3 puisque le solveur est appelé 3 fois pour chaque valeur de $k$. Le modèle ReqMod semble le plus robuste puisqu'il est capable de produire une solution jusqu'à $k = 39$ requêtes, contrairement à SomMod qui s'arrête à $k = 33$ requêtes. La figure 9a montre le temps moyen de résolution sur les 3 appels pour les deux modèles. Le temps de calcul reste relativement bas pour tous les valeurs de $k$, la plupart du temps en dessous de 25 secondes. Cela montre que le solveur soit trouve rapidement des solutions, soit conclut à la non-réalisabilité de l'instance en fonction de la difficulté. Un autre axe d'analyse porte sur le nombre de transferts, montré dans la figure 8c. Les deux modèles produisent des solutions utilisant assez peu le transport en commun, surtout quand il y a beaucoup de requêtes. Notons que SomMod semble plus facilement intégrer les transports en commun que ReqMod.

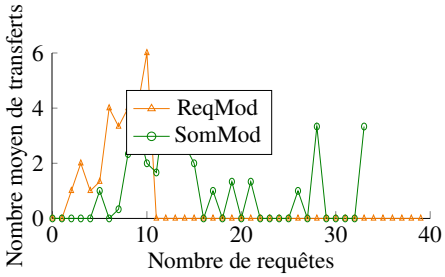
Comme notre but est de générer des solutions initiales et que l'on veut favoriser l'utilisation des transports en commun, on souhaiterait que les solution produites utilisent les transferts vers les trams pour qu'on puisse par la suite les optimiser avec d'autres méthodes. Dans la phase




(a) Nombre de succès selon le nombre de requêtes




(b) Temps moyen de résolution selon le nombre de requêtes




(c) Nombre moyen de transferts selon le nombre de requêtes

FIGURE 8 – Résultats dans le cas normal

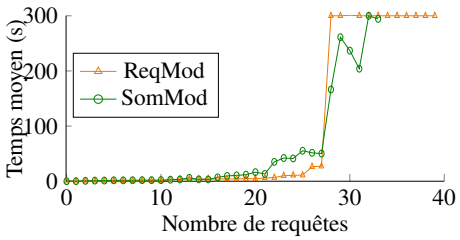
suivante d'optimisation, cela permettrait de ne pas avoir à couper les trajets des requêtes pour insérer du transport en commun. Ce faisant nous pourrons alors zoner les véhicules à la demande. Pour étudier cela, nous avons ajouté aux modèles des contraintes qui obligent chaque requête à changer de véhicule lors de l'utilisation du tram. Nous avons aussi ajouté une fonction objectif consistant à maximiser le nombre total de transferts.

Le nombre de succès du solveur après modification de chaque modèle est identique à la figure 8a. Ainsi, la fonction objectif n'influe pas sur la robustesse des modèles. Le temps moyen de résolution est représenté dans la figure 9a. Lorsqu'il est inférieur à 300 secondes, cela signifie que le solveur a trouvé une solution et qu'on est certain qu'il n'existe pas de solution avec un plus grand nombre de transferts pour au moins un des 3 appels. Lorsque le temps est de 300 secondes, cela signifie que le solveur a trouvé une ou des solutions, sans garantie d'avoir atteint le nombre maximum de transferts. Le solveur a donc continué de chercher jusqu'à la limite des 300 secondes pour les 3 appels. Pour ReqMod, on voit qu'il trouve facilement le nombre maximum de requêtes jusqu'au seuil de $k = 27$ requêtes. Passé ce seuil, il n'y arrive plus du tout. En revanche, SomMod ne possède pas ce seuil et il arrive à trouver des solutions de temps en temps au-delà de 27 requêtes. On peut le voir dans la figure 9b où les résultats de SomMod et ReqMod divergent au-delà de $k = 27$.

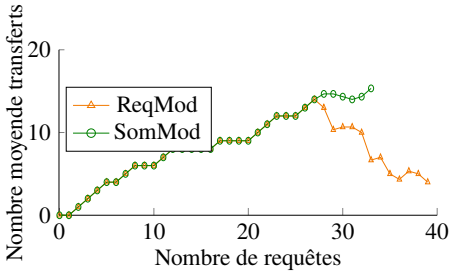
## 7. Conclusion

La question de la mobilité urbaine est un enjeu de société important. Elle est à la fois source et conséquence de profondes inégalités sociales. Le modèle actuel est également marqué par une prédominance des véhicules individuels, ce qui génère des impacts écologiques importants. Le transport multi-modal intégrant du transport à la demande est un moyen d'apporter de la flexibilité tout en utilisant de grandes lignes de transport en commun.

Ce problème est compliqué à résoudre comme l'ont montré les travaux dans la littérature. Pour obtenir de bonnes solutions, il est préférable d'optimiser des solutions initiales à l'aide de




(a) Temps moyen de résolution selon le nombre de requêtes




(b) Nombre moyen de transferts selon le nombre de requêtes

FIGURE 9 – Résultats avec la maximisation du nombre de transferts

métaheuristiques. Nous avons montré dans cet article comment produire des solution initiales sur un problème fortement contraint en faisant appel à la Programmation Par Contraintes. Deux modèles de PPC, SomMod et ReqMod ont été développés et testés sur une instance construite à partir de la ville du Havre. Les résultats ont montré que ReqMod est plus robuste en terme de succès, il arrive à fournir des solutions pour un certains nombre de requêtes là où SomMod n'y arrive pas. En revanche, SomMod semble plus efficace pour trouver des solutions avec transferts sur un certains nombre de requêtes. Si l'on veut trouver des solutions initiales pour une instance compliquée, avec beaucoup de requêtes par véhicules, il vaut mieux utiliser d'abord ReqMod, car ce modèle a plus de chance de fournir un résultat. SomMod peut être appelé ensuite, car il trouve des solutions plus intéressantes que ReqMod dans certains cas. Ils sont alors utilisés dans une approche en deux étapes.

Ces deux modèles sont suffisamment flexibles pour pouvoir être adaptés à une grande variété d'extensions du modèle, par exemple l'inclusion des horaires de passage des trams, ou la possibilité pour un utilisateur de commencer ou finir son parcours à pied. Certains articles ont également abordé la prise en compte de différences classes de passagers, avec des des restrictions de mobilité limitant par exemple le choix du mode de transport.

Enfin, ces modèles donnent une solution initiale à un problème d'optimisation. Il reste ensuite à optimiser les solutions fournies par une recherche locale. L'objectif est donc de coupler la Programmation Par Contraintes avec les métaheuristiques. On peut aussi tenter de modifier les modèles afin de favoriser certaines bonnes caractéristiques de solutions afin que la phase de recherche locale soit la plus efficace possible.

## Références

Bauchinger, L., Reichenberger, A., Goodwin-Hawkins, B., Kobal, J., Hrabar, M., and Oedl-Wieser, T. (2021). Developing sustainable and flexible rural–urban connectivity through complementary mobility services. *Sustainability*, 13(3) :1280.

Berbeglia, G., Pesant, G., and Rousseau, L.-M. (2011). Checking the feasibility of dial-a-ride instances using constraint programming. *Transportation Science*, 45(3) :399–412.

Cappart, Q., Thomas, C., Schaus, P., and Rousseau, L.-M. (2018). A constraint programming approach for solving patient transportation problems. In *Principles and Practice of Constraint*

*Programming : 24th International Conference, CP 2018, Lille, France, August 27-31, 2018, Proceedings 24*, pages 490–506. Springer.

Delecluse, A., Schaus, P., and Van Hentenryck, P. (2022). Sequence variables for routing problems. In *28th International Conference on Principles and Practice of Constraint Programming (CP 2022)*, pages 19 :1–19 :17. Schloss Dagstuhl-Leibniz-Zentrum für Informatik.

Deleplanque, S. and Quilliot, A. (2013). Transfers in the on-demand transportation : the darpt dial-a-ride problem with transfers allowed. In *Multidisciplinary international scheduling conference : theory and applications (MISTA)*, pages 185–205.

Häll, C. H., Andersson, H., Lundgren, J. T., and Värbrand, P. (2009). The integrated dial-a-ride problem. *Public Transport*, 1 :39–54.

Ham, A. (2023). Dial-a-ride problem : mixed integer programming revisited and constraint programming proposed. *Engineering Optimization*, 55(2) :257–270.

Laborie, P. and Rogerie, J. (2008). Reasoning with conditional time-intervals. In *FLAIRS conference*, pages 555–560.

Liaw, C.-F., White, C. C., and Bander, J. (1996). A decision support system for the bimodal dial-a-ride problem. *IEEE Transactions on Systems, Man, and Cybernetics-Part A : Systems and Humans*, 26(5) :552–565.

Liu, C., Aleman, D. M., and Beck, J. C. (2018). Modelling and solving the senior transportation problem. In *Integration of Constraint Programming, Artificial Intelligence, and Operations Research : 15th International Conference, CPAIOR 2018, Delft, The Netherlands, June 26–29, 2018, Proceedings 15*, pages 412–428. Springer.

Molenbruch, Y., Braekers, K., Hirsch, P., and Oberscheider, M. (2021). Analyzing the benefits of an integrated mobility system using a matheuristic routing algorithm. *European Journal of Operational Research*, 290(1) :81–98.

Posada, M., Andersson, H., and Häll, C. H. (2017). The integrated dial-a-ride problem with timetabled fixed route service. *Public Transport*, 9 :217–241.

Posada, M. and Häll, C. H. (2020). A metaheuristic for evaluation of an integrated special transport service. *International Journal of Urban Sciences*, 24(3) :316–338.

Potter, B. C. (1976). Ann arbor's dispatching system. *Transportation Research Record*.

Thomas, C., Kameugne, R., and Schaus, P. (2020). Insertion sequence variables for hybrid routing and scheduling problems. In *International Conference on Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, pages 457–474. Springer.

Wilson, N. H. M., Weissberg, R. W., and Hauser, J. (1976). Advanced dial-a-ride algorithms research project. Technical report, Massachusetts Institute of Technology, Department of Materials Science and Engineering.

## Biographie des auteurs

Louise Penz est en doctorat en Informatique à l'Université Le Havre Normandie depuis 2021, au sein du laboratoire LITIS (Laboratoire d'Informatique, de Traitement de l'Information et des Systèmes). Elle est encadrée par Éric Sanlaville et Christophe Duhamel. Lors de ses études, elle s'est spécialisée en Recherche Opérationnelle dans le master ORCO (Operations Research, Combinatorics and Optimization) à l'université Grenoble Alpes. Sa thèse porte sur le transport de passagers dans une ville et ses alentours. Deux points majeurs sont abordés : la construction de solutions ainsi que l'optimisation des routes selon certaines contraintes.

Éric Sanlaville est professeur en Informatique à l'Université Le Havre Normandie. Il est co-directeur du LITIS (Laboratoire d'Informatique, de Traitement de l'Information et des Systèmes). Spécialiste en Recherche Opérationnelle, il a longtemps travaillé sur les problèmes d'ordonnancement avant de se tourner vers la logistique, essentiellement portuaire. Il s'intéresse à la conception et l'analyse d'algorithmes d'optimisation, souvent en présence d'incertitudes, en utilisant des modèles de graphes, en particulier dynamiques (dépendant du temps). Il publie dans les revues majeures de la RO et du transport.

Christophe Duhamel est enseignant-chercheur en Informatique à l'Université Le Havre Normandie. Il est membre du LITIS (Laboratoire d'Informatique, de Traitement de l'Information et des Systèmes). Ses travaux de recherche s'inscrivent en Recherche Opérationnelle (RO) et portent essentiellement sur l'optimisation des problèmes de transport et des problèmes de conception et de routage dans les réseaux. Dans le cadre du transport, il s'est intéressé aussi au transport de biens comme à celui de personnes, avec la prise en compte de critères de Qualité de Service ou d'aspects multimodaux. Ses travaux se concentrent sur la proposition de modèles mathématiques et l'application de méthodes d'optimisation exacte ou approchées (heuristiques et métaheuristiques). Il a publié plus de 30 articles dans des revues internationales.