UNIVERSITÉ DU QUÉBEC À MONTRÉAL

PLANIFICATION COOPÉRATIVE EN TEMPS RÉEL D'ITINÉRAIRES D'UNE FLOTTE DE VÉHICULES ÉLECTRIQUES PARTAGEANT DES BORNES DE RECHARGE

MÉMOIRE

PRÉSENTÉ

COMME EXIGENCE PARTIELLE

DE LA MAÎTRISE EN INFORMATIQUE

PAR

MARC-ANDRÉ LAVOIE

DÉCEMBRE 2022

REMERCIEMENTS

Je désire remercier mon directeur, Éric Beaudry, professeur au Département d'informatique de l'Université du Québec à Montréal (UQAM). Il a fait preuve d'une patience exceptionnelle en me guidant minutieusement au travers de la sélection de mon sujet ainsi que du perfectionnement de l'idée de ce mémoire. Je veux également remercier ma conjointe Marjolaine qui a su me soutenir dans cette aventure au même moment où nous élevons trois jeunes enfants. Finalement, merci au Cégep du Vieux-Montréal pour son soutien logistique et financier.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

RÉSUMÉ

Ce mémoire aborde le problème de planification coopérative d'itinéraires pour une flotte de véhicules électriques (VÉ) utilisant un réseau de bornes de recharge.

Les réseaux de bornes actuels n'ont généralement pas de système de réservation de bornes. Dans les outils d'aide la planification d'itinéraire de VÉ, chaque VÉ est planifié de façon indépendante. Au mieux, le temps d'attente espéré, basé sur l'historique d'utilisation des bornes, peut être intégré. Planifier chaque VÉ de façon indépendante peut être qualifié de planification non coopérative puisque chaque VÉ utilise les bornes permettant le meilleur itinéraire. Lorsque plusieurs VÉ coexistent, cela peut mener à une utilisation non optimale des bornes. Par exemple, deux VÉ partant en même temps d'une même origine vers une même destination voudront utiliser les mêmes bornes en même temps.

Dans ce mémoire, nous proposons des algorithmes de planification coopérative d'itinéraire de VÉ. Ces algorithmes ont pour objectif d'optimiser l'usage global des bornes. Une propriété de ces solutions est que certains VÉ peuvent emprunter des itinéraires légèrement plus coûteux en temps. Par exemple, des VÉ peuvent faire de petits détours pour éviter les conflits ou la surutilisation de certaines bornes. Les travaux réalisés s'inspirent de travaux réalisés en planification coopérative de chemins pour des groupes d'agents dans des jeux vidéo.

Les planificateurs coopératifs les plus simples exécutent un planificateur individuel pour chaque VÉ mais dans des ordres différents. Chaque ordre d'exécution peut générer un plan global différent. Une modification de cette approche est de calculer successivement les plans par morceau : le planificateur coopératif simple trouve meilleur plan dans un intervalle de temps compris entre 0 et τ (τ étant l'horizon temporel). Ensuite, le meilleur plan dans l'intervalle τ à 2τ est calculé et ainsi de suite jusqu'à ce que tous les VÉ soient à destination. Cet algorithme élargit l'espace de plan exploré tout en rendant le planificateur utilisable en temps réel. Finalement, des méthodes originales avec réparation locale de solutions existantes sont proposées. Ces derniers éliminent les pires attentes aux bornes de recharge (en tenant compte des détours) en explorant récursivement des solutions voisines.

Les algorithmes ont été évalués sur des scénarios artificiels générés avec des données réelles comme la carte routière du Québec extraite d'OpenStreetMap et du réseau de bornes du Circuit électrique.

INTRODUCTION

Pour lutter contre les changements climatiques, une transition énergétique s'impose. Dans le domaine du transport de personnes, les véhicules électriques (VÉ) sont appelés à remplacer les véhicules traditionnels à essence et au diésel. Plusieurs pays, états et provinces, dont le Canada et le Québec, ont annoncé l'interdiction de la vente de véhicules neufs à essence d'ici 2035.

La transition aux VÉ pose plusieurs défis importants. Alors qu'un véhicule conventionnel est équipé d'un simple réservoir à essence, les VÉ doivent pouvoir stocker de l'énergie facilement transformable en électricité. Actuellement, les VÉ les plus répandus sur le marché sont équipés de batteries de type lithium-ion. Ce type de batterie a de nombreux inconvénients face à un réservoir à essence conventionnel comme le coût de fabrication, la faible densité énergétique (énergie par volume ou masse), la durée de recharge, etc. Les véhicules à pile à combustible d'hydrogène présentent également une alternative écologique aux véhicule à essence. Toutefois, ces derniers présentent encore d'autres inconvénients majeurs en raison des défis de la production, de la compression et de la liquéfaction de l'hydrogène.

Bien que la technologie des batteries lithium-ion se soit améliorée au cours de la dernière décennie, l'autonomie et la durée de recharge des VÉ demeurent des contraintes importantes qui freinent leur adoption massive. À titre d'exemple, la Nissan Leaf 2022 en version de base, l'un des modèles de VÉ les plus abordables sur le marché, n'offre que 363 km d'autonomie dans des conditions idéales et nécessite 30 minutes pour se recharger à 80% de sa capacité. Pour ces raisons, les déplacements sur de longues distances sont plus compliqués en raison des arrêts

requis à des bornes de recharge. La figure 0.1 montre un exemple de trajet de Montréal à Matane d'un véhicule de modèle Nissan Leaf 2010 avec une autonomie de 242 km complètement chargée au départ. La durée de déplacement est de 6h16m, mais la durée totale des deux recharges (1h) et d'attente aux bornes de recharge (57m) portent le total du trajet à 8h13m.

De plus, les VÉ sur le marché sont aussi généralement soit petits, donc peu convenables pour les besoins d'une famille, ou dispendieux.

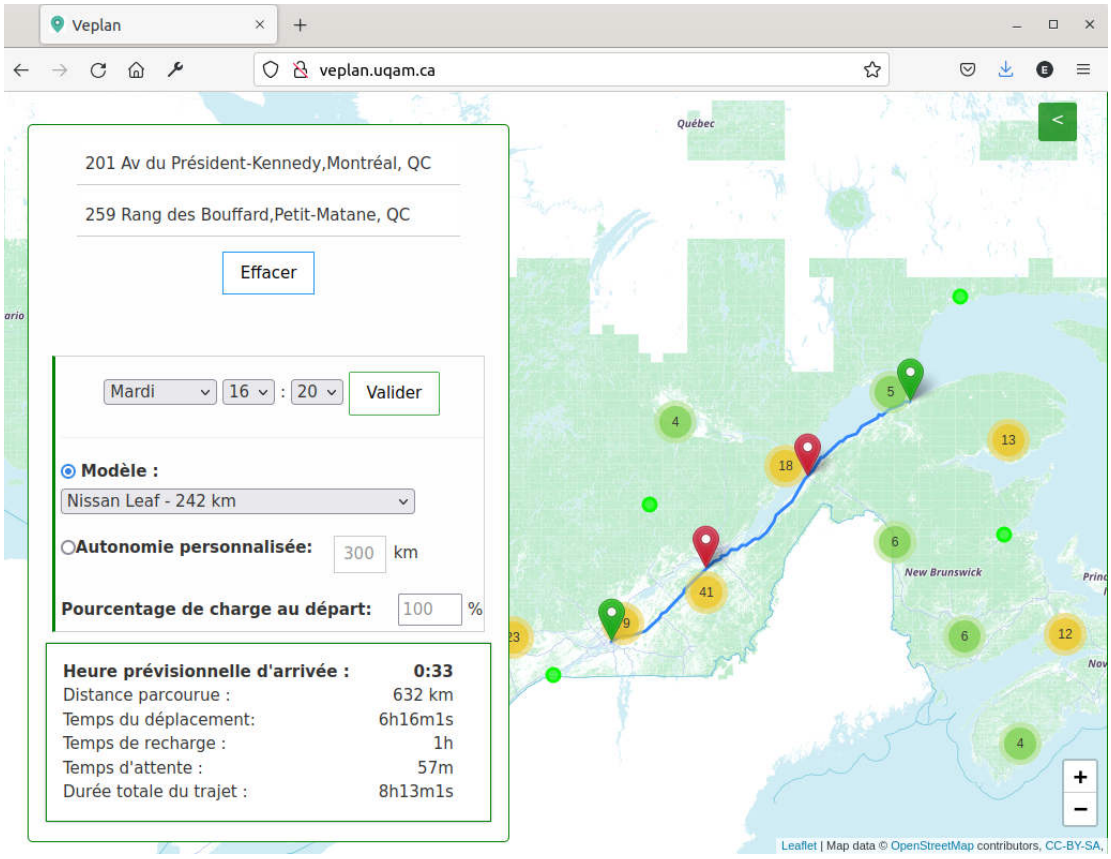


Figure 0.1 Exemple d'itinéraire calculé par VE Plan (Jaël Gareau et al., 2022)

Pour aider les utilisateurs de VÉ à planifier leurs trajets, plusieurs systèmes d'aide à la décision existent. Une équipe du laboratoire CRIA de l'UQAM a développé

l'application interactive de planification de déplacement en véhicule électrique VE Plan (Jaël Gareau *et al.*, 2022). Ce dernier permet de planifier à un usager de planifier un déplacement en spéficiant les points de départ et de destination, le modèle de VÉ ainsi que la charge de la batterie au départ. L'itinéraire est construit suites aux travaux sur les VÉs.

Le Circuit électrique, division d'Hydro-Québec chargée de développer et d'opérer un réseau public de bornes électriques, a aussi déployé un assistant de planification dans son application mobile. D'autres sites Web existent, dont ChargeHub (ChargeHub, 2021).

Le nombre de VÉ sur les routes a considérablement progressé dans les cinq dernières années. Le nombre de bornes de recharge rapide à courant continu (BRCC) a aussi progressé, mais à un rythme moindre que le nombre de VÉ. Par exemple, au Québec, le nombre de VÉ a franchi la barre symbolique de 100 000 en 2021 [1] (figure 0.2), et le nombre de bornes rapides à courant continu (BRCC) a atteint 450 [2].

---

1. Selon un rapport détaillant la situation des VÉ au Québec selon l'Association des Véhicules Électriques au Québec (Association des Véhicules Électriques du Québec, 2021)

2. Selon un rapport de Véhicule Électrique Québec (Énergie et ressources naturelles Québec, 2021)

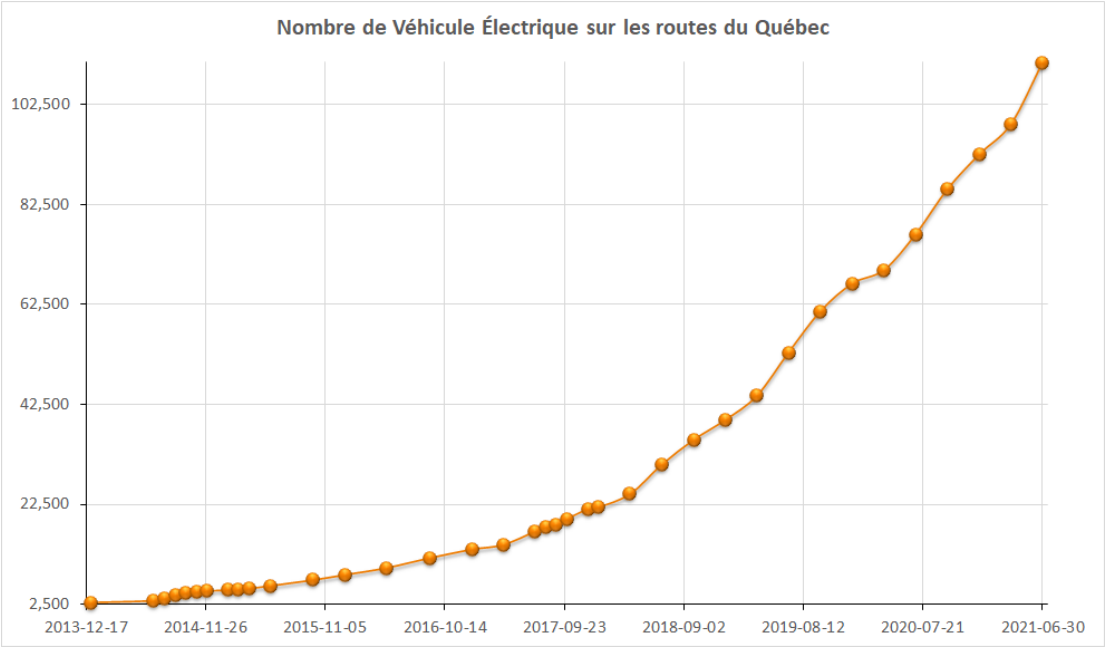


Figure 0.2 Évolution du nombre de VÉs immatriculés dans la province de Québec.

Ainsi, les bornes de recharge sont de plus en plus utilisées. L'augmentation du taux d'occupation des bornes de recharge implique que des usagers doivent parfois attendre plus longtemps en file avant de pouvoir y brancher leur VÉ.

Les applications mobiles permettant de consulter l'état d'occupation des bornes n'atténuent ce problème que partiellement (Le Circuit Électrique, 2021). Lorsqu'un conducteur de VÉ sait d'avance qu'une borne est occupée ou endommagée, il peut décider de faire un détour pour se rendre à une borne différente. Un historique des données d'occupation peut permettre à un planificateur de considérer le temps d'attente espéré aux bornes pour minimiser l'effet d'attente (Gareau *et al.*, 2019).

À mesure que la densité énergétique des batteries s'améliore, l'autonomie des VÉ augmentera et chaque VÉ a moins souvent besoin d'être rechargé. La puissance

de recharge augmente également, ce qui signifie des temps de recharge diminué et une réduction de l'achalandage aux bornes de recharge. Le problème d'attente aux bornes de recharge est pertinent donc d'être étudié pour répondre à un besoin qui persistera encore au moins cinq à dix ans.

Une solution complémentaire pour gérer l'attente aux bornes est d'introduire un système de réservation. Un utilisateur peut, de cette manière, réserver des bornes à des heures précises pour garantir leurs disponibilités. Bien entendu, en pratique, il n'y a pas de garanties qu'un autre usager n'emprunte pas une borne déjà réservée, mais la conduite automatique de véhicule pourra aider à pallier à ce problème.

Dans ce mémoire, nous proposons des algorithmes de planification coopératifs permettant de gérer, de façon centralisée, une flotte de VÉ en considérant. L'objectif de ces algorithmes est de calculer des itinéraires ayant un coût global inférieur comparativement à la situation où chaque VÉ planifie son itinéraire indépendamment.

Les méthodes de planification présentées dans ce mémoire s'inspirent de travaux en planification de chemin coopérative. Le reste du mémoire est structuré comme suit. Le chapitre 1 détaille les hypothèses de notre modélisation et formalise mathématiquement les concepts. Le chapitre 2 passe en revue la littérature pour les planificateurs individuels de VÉ de même que des techniques générales de planification coopérative de chemins. Le chapitre 3 introduit les planificateurs hors ligne. Le chapitre 4, généralise les travaux aux planificateurs dits en ligne (temps réel). Le chapitre 5 précise la méthodologie des expériences et le chapitre 6 présente les résultats.

CHAPITRE I

PRÉSENTATION DU PROBLÈME

Ce chapitre introduit les hypothèses, la formalisation mathématique et les fonctions objectif. Cette fondation est nécessaire avant d'aborder pleinement les planificateurs.

1.1 Hypothèses

Pour simplifier le problème, nous introduisons des hypothèses. Pour commencer, nous fixons des hypothèses contraignantes. Au fil du mémoire, nous en assouplirons quelques-unes afin de considérer des cas plus complexes et générer de meilleurs plans.

1.1.1 Déterminisme

Une hypothèse forte de la modélisation est celle du déterminisme. Celui-ci se manifeste à plusieurs niveaux qui sont détaillés dans les sous-sections ci-dessous.

Vitesse moyenne des VÉ

La vitesse moyenne des véhicules est considérée à $\upsilon = 72$ km/h (20m/s), sur tous les segments de route. L'accélération et la décélération sont instantanées, c'est à dire ignorées.

On ne considère pas les retards pouvant être causés par de multiples raisons comme : prendre une pause sur un long trajet, conditions météo affectant les conditions routières, présence de congestion, pour éviter la présence de travaux sur la route, etc.

Cette hypothèse réduit la complexité des planificateurs sans compromettre leurs validités : une vitesse moyenne qui traduit un temps de déplacement équivalent

sur une arête quelconque existe. Cette vitesse moyenne de déplacement peuvent varier dans le temps, selon la présence de construction, présence de congestion, évènement spécial etc.

Durée des actions aux bornes de chargements

Le branchement et le débranchement à une borne sont réputés instantanés dès qu'un VÉ est arrivé et que la borne est disponible. Lorsque le chargement est complet, le VÉ quitte immédiatement pour sa prochaine destination. Les bornes sont fonctionnelles en tout temps.

S'il y a présence d'une file d'attente, il n'y a pas de délai entre le débranchement du VÉ sortant et le branchement du prochain VÉ de la file d'attente : dès qu'un a terminé, le suivant est branché.

Dans la pratique, il serait possible d'introduire une estimation du délai en intégrant une constante pour le branchement et le débranchement. Cela n'a pas d'impact sur la validité des algorithmes présentés dans ce mémoire.

Autonomie des VÉ

Dans un contexte réaliste, l'autonomie varie selon plusieurs facteurs comme : modèle de VÉ, l'âge de la batterie, la vitesse (celle-ci affectant la force de traînée), le dénivelé de la route, les conditions météorologiques et de nombreux autres facteurs (Yang *et al.*, 2014). Une fois de plus, cette simplification n'invalide pas l'étude des planificateurs présentés : pour chacun des véhicules qui s'insèrent sur la route, une capacité maximale et une autonomie peut être estimés selon les facteurs énumérés (voir (Yang *et al.*, 2014), (Abousleiman et Rawashdeh, 2015) et (Fiori *et al.*, 2016) pour des modèles). Cela simplifie la modélisation car le coût entre chaque

paire de bornes peut être considéré comme constant en énergie.

Ce mémoire propose un modèle simplifié d'autonomie qui dépend uniquement du modèle de VÉ (voir la table 5.1). Cependant, la consommation est constante et identique pour tous les modèles de VÉs.

Unités

En pratique on exprime l'autonomie de la batterie en kWh et celle d'un VÉ, en km. Ainsi, à vitesse constante, la consommation s'exprime selon le rapport de ces deux quantités, soit des $\frac{\mathrm{kWh}}{\mathrm{km}} = \frac{\mathrm{Wh}}{\mathrm{m}}$.

Chargement de la batterie

Le puissance de chargement d'une batterie est exprimé en kW et celle d'un VÉ, en $\frac{\mathrm{km}}{\mathrm{h}}$ (ou, en utilisant le SI, en $\frac{\mathrm{m}}{\mathrm{s}}$). Le taux de chargement η d'un VÉ équipé d'une batterie de type lithium-ion est constant à 139m/s. Ce taux réplique le taux de chargement moyen d'un modèle Hyundai Ioniq qui se branche à une borne de recharge rapide de 50 kW de 20% à 80% (se référer au simulateur de charge (Automobile Propre, 2021)).

Dans la réalité, le temps de chargement dépend en réalité du type de borne, varie selon le modèle du véhicule, la puissance de la borne, la charge de la batterie, le taux d'occupation de la grille d'électricité, de l'usure de la batterie et bien d'autres facteurs (Hoke *et al.*, 2011). Dans ce mémoire, uniquement les bornes rapides sont considérées pour des fins de simplification.

Dans les faits, le taux de chargement est typiquement non linéaire : lorsque la batterie est vide, elle se charge rapidement. La puissance décroît rapidement lorsqu'on s'approche d'une charge complète. Un exemple de la puissance de chargement dérivée est illustrée sur la figure 1.1.

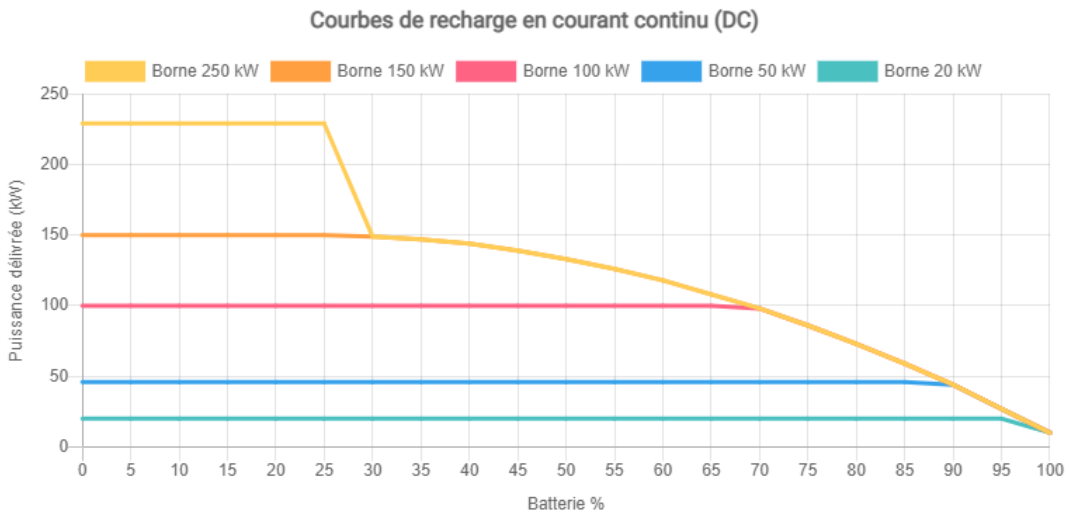


Figure 1.1 Exemple de la puissance de chargement dérivée pour le modèle Hyundai Ioniq 5 selon le niveau de charge de la batterie et la puissance de borne. La puissance dérivée étant plus grande lorsque la batterie est presque vide. Source : (Automobile Propre, 2021).

État de la charge de la batterie

La batterie est entièrement chargée lorsqu'un véhicule débute sa route et chaque fois qu'il quitte une borne. Le véhicule bénéficie ainsi toujours de son autonomie maximale à chaque départ. De plus, cela signifie qu'il chargera complètement la batterie à chaque borne (aucune charge partielle possible).

Nous ne nous préoccupons pas du niveau de charge lorsque le VÉ arrive à destination. En d'autres mots, un VÉ arrivé à destination n'a pas besoin d'être rechargé et donc ne bloque la borne.

Buts constants

L'absence d'incertitude dans la destination (finale ou intermédiaire) d'un VÉ de la flotte est une forme de déterminisme. Le VÉ s'engage à suivre un itinéraire qui ne change pas en cours de route. Dans un scénario réel, un usager de la route peut changer d'idée à cause d'une multitude de facteurs : un arrêt soudain pour un café ou la salle de bain, un changement d'intention encore un usager peut délibérément ignorer le chemin proposé par un système de planification.

1.2 Formalisation

La notation mathématiques est maintenant présentée.

1.2.1 Véhicules et carte routière

Une flotte de véhicules électriques V est un ensemble $V = \{v_1, v_2 \ldots, v_{|V|}\}$. Chaque VÉ de la flotte circule sur une carte routière qui est un graphe orienté $G(S, A)$. Les sommets S sont un ensemble noté $S = \{s_1, s_2, \cdots, s_{|S|}\}$ et représentent un point (croisement de routes) sur la carte routière.

Les arêtes $A = \{a_1, a_2, \cdots, a_{|A|}\}$ sont orientées et représentent un segment de route sur la carte routière. Une arête $a = (s_i, s_j) \in A$ (avec s_i et $s_j \in S$) lie deux sommets. Autrement dit, chaque arête est liée par deux carrefours. Un point peut lier plusieurs arêtes. Le poids d'une arête est donné par la fonction $W(a) \in \mathbb{R}_*^+$ et représente la distance, en mètres, de l'arête a.

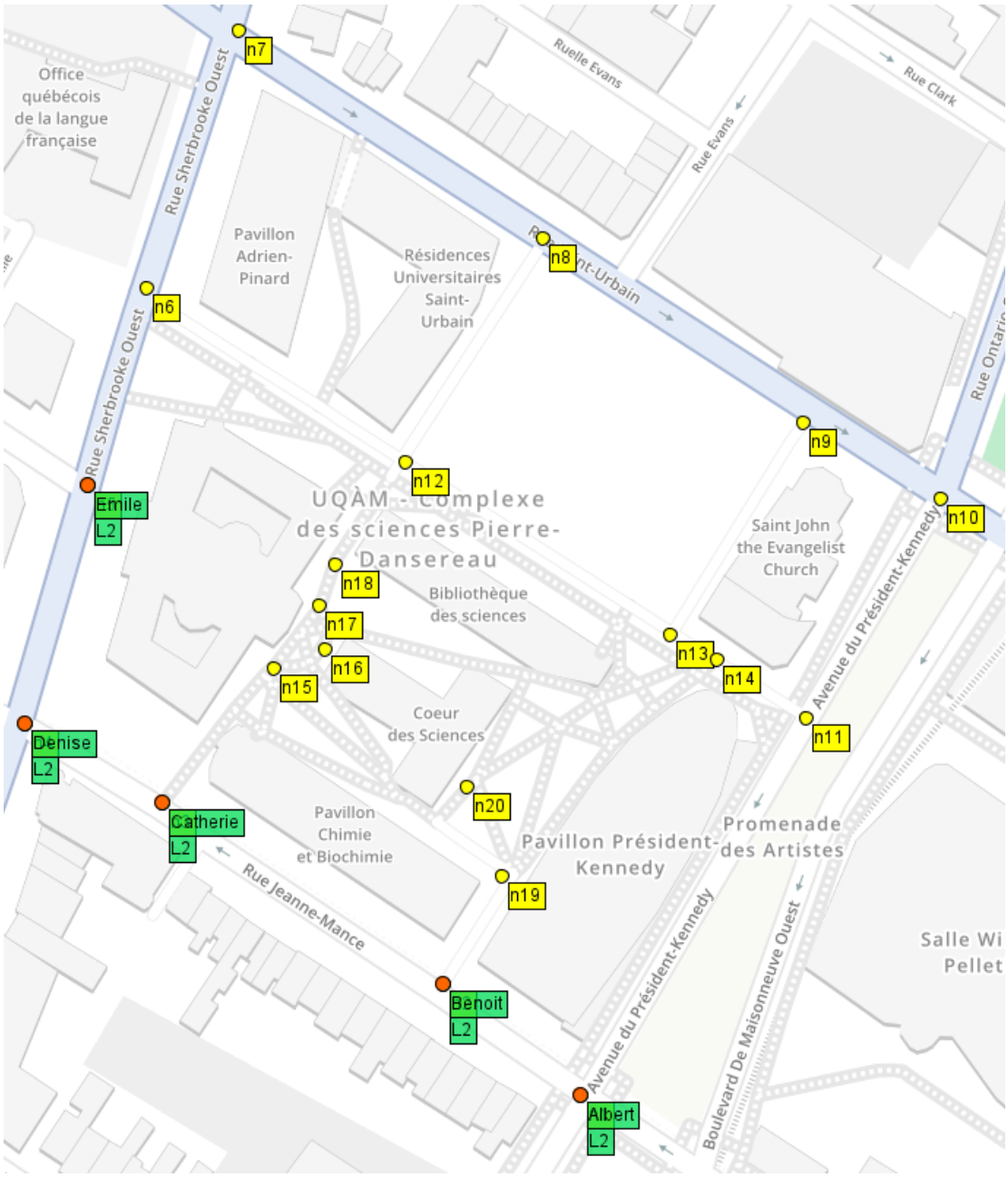


Figure 1.2 Carte OpenStreetMap du Coeur des sciences de l'UQÀM avec des bornes de recharge fictives (en orange et étiquette en vert). Les sommets sans borne sont indiqués en jaune.

L'itinéraire routier d'un VÉ v_i est une suite de sommets notée $C(v_i) = \langle c_o, c_1, c_2 \cdots, c_d \rangle$

avec $c_i \in S$ et c_o,c_d qui sont, respectivement, les noeuds de départ et de destination de v_i. La distance de déplacement entre deux sommets s'exprime

$$W(s_o, s_d) = \sum_{k=1}^{K-1} W(s_k, s_{k+1}) \tag{1.1}$$

où les s_k correspondant aux points intermédiaires parcourus sur la carte pour aller de s_o à s_d et K le nombre de points parcourus. Cette distance est convertible en temps de déplacement $d_{déplacement}(s_o, s_d)$ en divisant la distance par la vitesse constante v.

$$d_{déplacement}(s_o, s_d) = \frac{W(s_1, s_{|K|})}{v} \tag{1.2}$$

1.2.2 Borne de recharge

À chaque $s \in S$ peut correspondre une seule borne de chargement indiquée par un booléen a_borne. Si plusieurs bornes sont présentes à un sommet s, un nouveau sommet avec la même position est créé pour chaque borne. Tout VÉ qui désire se charger va occuper une borne pour la durée du chargement uniquement (hypothèse 1.1.1). Si un véhicule veut se charger mais qu'aucune station n'est disponible à une borne donnée, celui-ci s'insère dans une file d'attente.

1.2.3 Réseau des bornes de recharge

En premier lieu, on définit un ensemble de bornes $B \subseteq S$ avec $B = \{b_1, b_2, \cdots, b_{|B|}\}$. B contient les sommets ayant une borne, c'est à dire $B \leftarrow s \in S|s.a\_borne$. Les arêtes de l'ensemble de bornes forment un graphe dont chaque sommet est connecté à tous les sommets à l'intérieur d'un rayon correspondant à l'autonomie d'un VÉ. Les arêtes sont notées :

$$A = \{\alpha_1, \alpha_2, \cdots, \alpha_{|A|}\} \tag{1.3}$$

Le graphe des bornes est ainsi défini comme $G_b = (B, A)$. Pour une arête $\alpha = (b_o, b_d) \in A$ du graphe des bornes, le poids associé $W(\alpha) \in \mathbb{R}_*^+$ est la distance de l'itinéraire le plus cours pour se passer de b_o à b_d sur la carte routière. Un exemple d'un tel graphe des bornes se trouve sur la figure 3.2. Le précalcul est présenté à la section 3.1.

1.2.4 Entrées et sorties des planificateurs

Les entrées et les sorties des planificateurs explorés dans ce mémoire étant identiques, elles sont présentés ici. Les symboles et la notation sont empruntés sont inspirées de (Ghallab *et al.*, 2016).

Chaque VÉ émet une unique *requête*. Une requête est un triplet composé de l'origine, la destination et le temps de départ $r_i = (s_o, s_d, \tau_i)$. L'ensemble des requêtes est une suite R ordonnée en ordre croissant selon le temps de départ.

$$R = \langle (\nu_{1,o}, \nu_{1,d}, \tau_1), (\nu_{2,o}, \nu_{2,d}, \tau_2), \cdots, (\nu_{N,o}, \nu_{N,d}, \tau_N) \rangle \tag{1.4}$$

Il y a *2 entrées* pour les planificateurs : un graphe des bornes G_b et une suite de requêtes R.

En sortie, les planificateurs génèrent un *plan de la flotte* dénommé *plan global* : c'est une suite de plans individuels de VÉ (itinéraires) π_i associés à chaque requêtes

r_i, i.e :

$$\begin{aligned}\pi &= \langle \pi_1, \pi_2, \cdots, \pi_N \rangle \\ &= \langle \langle s_{1,o}, s_{1,1}, \cdots, s_{1,d} \rangle, \langle s_{2,o}, s_{2,1}, \cdots, s_{2,d} \rangle, \cdots, \langle s_{|V|,o}, s_{|V|,1}, \cdots, s_{|V|,d} \rangle \rangle\end{aligned} \tag{1.5}$$

Une requête peut ne pas avoir de solutions, par exemple quand un VÉ est en dehors du circuit ou qu'il est impossible de créer un itinéraire sur le graphe des bornes. Dans les expérimentations, toutes les requêtes ont une solution associée.

Un plan global est complet si tous les agents sont arrivés à leurs destination respective. Dans le cas contraire, le plan global est dit partiel. L'espace de tous les plans globaux possibles pour une flotte est noté Π.

1.2.5 Fonctions objectif

Analoguement aux problèmes de planification de chemin multi agent, la sélection d'une fonction objectif n'est pas triviale : plusieurs fonctions objectif $M(\pi)$ existent et chacune présente des propriétés différentes (Sharon *et al.*, 2015). Avant d'en étudier quelques unes, un préambule est requis.

Durée de déplacement sur une arête

Sur le graphe des bornes, la durée de trajet d'une arête $\alpha = (b_o, b_d))$ est définie comme

$$M_{\text{DuréeTotale}}(\alpha) = d_{\text{déplacement\_od}} + d_{\text{attente\_d}} + d_{\text{charge\_d}} \tag{1.6}$$

avec...

- $d_{\text{déplacement\_od}}$ la durée de déplacement entre b_o et b_d ;
- $d_{\text{attente\_d}}$ la durée de l'attente avant le chargement à b_d, si applicable ;

— d_{charge_d} la durée de chargement à b_d.

Les hypothèses de vitesse, consommation et taux de chargement constants jumelée à la garantie d'avoir une batterie pleine à chaque arrêt permettent de calculer directement d_{charge_d} à partir de $d_{déplacement\_od}$.

$$d_{charge\_d} = \frac{d_{déplacement\_od} \times v}{\eta}$$

$$d_{charge\_d} = \frac{d_{déplacement\_od} \times 20 m/s}{139 m/s}$$

$$d_{charge\_d} = 0.144 \times d_{déplacement\_od}$$

$$d_{charge\_d} = \psi \times d_{déplacement\_od}$$

On définit la constante $\psi = 0.144$ qui est sans unité. Elle représente le ratio entre la durée de chargement et la durée de déplacement. Ainsi, l'équation (1.6) peut s'exprimer comme suit :

$$M_{DuréeTotale}(\alpha) = (1 + \psi) d_{déplacement\_od} + d_{attente\_d} \tag{1.7}$$

Durée de déplacement d'itinéraire

Soit le plan local π_i d'un VÉ (notation inspirée de (Ghallab *et al.*, 2016) correspond à son itinéraire. La durée de ce plan local est la somme des durées de toutes les arêtes que le VÉ traverse, i.e.

$$M_{DuréeTotale}(\pi_i) = \sum_k M_{DuréeTotale}(\alpha_k) \tag{1.8}$$

En admettant qu'une solution existe, il existe toujours au moins un plan local optimal $M^*_{DuréeTotale}(\pi_i)$ tel qu'il est impossible d'en réduire la durée davantage.

$$M^{*}_{\text{DuréeTotale}}(\pi_i) \leq M_{\text{DuréeTotale}}(\pi_i) \tag{1.9}$$

C'est le plan que le VÉ emprunterait s'il était le seul usager de la route. Il n'aura donc aucune attente aux bornes ou détour à effectuer sur la route.

Finalement, la pénalité est introduite :

$$M_{\text{Pénalité}}(\pi_i) = \frac{M_{\text{DuréeTotale}(\pi_i)}}{M_{\text{DuréeTotale}^{*}(\pi_i)}} \in [1, \infty) \tag{1.10}$$

C'est le rapport entre la durée d'un plan local et la durée du plan local optimal telle que définie à l'équation (1.9).

Six fonctions objectifs sont proposées pour évaluer les plans globaux.

Durée d'itinéraire la plus longue

Comme première fonction objectif pour évaluer un plan global, définissons la durée de plan local la plus longue :

$$M_{\text{DuréeMaximum}}(\pi) = \max\{M_{\text{DuréeTotale}}(\pi_1), \cdots, M_{\text{DuréeTotale}}(\pi_N)\} \tag{1.11}$$

Minimiser cette fonction est une fonction objectif adéquate : la pire durée de déplacement sera systématiquement améliorée.

Il y a cependant deux problèmes avec $M_{\text{DuréeMaximum}}(\pi)$: il pourrait avoir une situation qu'il serait avantageux d'augmenter le temps de déplacement d'un seul véhicule pour réduire la durée de déplacement de tous les autres. Cette fonction objectif ne créée pas cet effet car elle se concentre sur le pire itinéraire.

De plus, cette fonction objectif permet surtout d'optimiser les itinéraires les plus long d'une flotte de VÉs. Soit un VÉ v_{long} a un itinéraire plus long que les autres, $M_{DuréeTotale}(\pi_{v_{long}})$ sera la meilleure valeur possible de la fonction objectif. La durée d'un v_{cours} ne sera pas considérée avec $M_{DuréeMaximum}(\pi)$.

Durée moyenne des itinéraires

Minimiser le total des durées des itinéraires (éq. 1.12) est une autre possibilité, mais la durée moyenne (éq. 1.13) a l'avantage d'être invariante par rapport au nombre de VÉ sur la route.

$$M_{DuréeTotale}(\pi) = \sum_{i=1}^{N} M_{DuréeTotale}(\pi_{v_i}) \tag{1.12}$$

$$M_{DuréeMoyenne}(\pi) = \frac{M_{DuréeTotale}(\pi)}{N} \tag{1.13}$$

L'ensemble des véhicules est représenté avec cette fonction objectif et non seulement le véhicule avec le pire itinéraire. Un problème persiste : un VÉ avec un plan très court pourrait faire un grand détour/attendre beaucoup afin d'accommoder tous les autres véhicules. Cette fonction objectif peut pénaliser un seul VÉ de manière importante par rapport à la durée de son itinéraire.

Moyenne de la somme des différences des carrés des durées

La différence du carré d'un itinéraire $\pi_i \in \pi$ est :

$$M_{DuréeDiffCarré}(\pi_i) = (M_{DuréeTotale}(\pi_i) - M_{DuréeMoyenne}(\pi))^2 \tag{1.14}$$

Elle permet de définir la *moyenne de la somme des différences des carrés des*

durées d'un plan global

$$M_{\text{MoyenDuréeDiffCarré}}(\pi) = \frac{\sum_i M_{\text{DuréeDiffCarré}}(\pi_{v_i})}{N} \tag{1.15}$$

Cette fonction objectif regroupe les itinéraires afin qu'ils aient une durée similaire, évitant ainsi qu'un VÉ soit pénalisé de manière trop importante par rapport à l'ensemble de la flotte. Les propriétés de linéarité sont perdues. Une remarque supplémentaire est que l'on pénalise identiquement v_{long} et v_{petit} pour des détours de même durée. Cela peut générer une solution qui doublerait le temps de trajet de v_{petit} mais affecterait moins v_{long}.

Pire pénalité

La pire pénalité est définie comme :

$$M_{\text{PirePénalité}}(\pi) = \max\{M_{\text{Pénalité}}(\pi_1), \cdots, M_{\text{Pénalité}}(\pi_N)\} \tag{1.16}$$

a l'avantage d'être invariante par rapport à la durée de l'itinéraire de chaque VÉ, ce qui règle le problème si on a v_{long}. Minimiser la pire pénalité partage un inconvénient de la pire durée : la portée de l'optimisation est limitée étant donné qu'un seul véhicule est optimisé à la fois.

Pénalité moyenne

La pénalité totale est

$$M_{\text{PénalitéTotale}}(\pi) = \sum_{i=1}^{N} M_{\text{Pénalité}}(\pi_i) \tag{1.17}$$

Pour la rendre invariante par rapport au nombre de véhicule, la pénalité moyenne est avantageuse.

$$M_{\text{PénalitéMoyenne}}(\pi) = \frac{M_{\text{PénalitéTotale}}(\pi)}{N} \tag{1.18}$$

Cette fonction objectif est invariante par rapport au nombre de voitures et à la durée des itinéraires. Elle comporte cependant encore le risque de pénaliser un VÉ beaucoup trop par rapport aux autres.

Moyenne du carré des pénalités

La différence du carré des pénalités d'un itinéraire $\pi_i \in \pi$ est :

$$M_{\text{PénalitéDiffCarré}}(\pi_i) = (M_{\text{Pénalité}}(\pi_i) - M_{\text{PénalitéMoyenne}}(\pi))^2 \tag{1.19}$$

La moyenne du carré des pénalités d'un plan global est définie comme

$$M_{\text{MoyenPénalitéDiffCarré}}(\pi) = \frac{\sum_i M_{\text{PénalitéDiffCarré}}(\pi_i)}{N} \tag{1.20}$$

Celle-ci évite qu'une petite partie des véhicules soient pénalisés de manière disproportionnée par rapport à la flotte. Dans le cas de ν_{long} et ν_{petit}, les deux plans seraient pénalisés en lien à la durée du trajet. Par contre, l'inconvénient des pertes des propriétés de linéarité sont également présentes. Cette propriété limite l'application de techniques de recherche opérationnelle, comme la méthode du simplexe, qui consiste à optimiser des systèmes d'équations linéaires (Karloff, 2009). Ces méthodes ne sont donc pas utilisées dans ce mémoire.

CHAPITRE II

ÉTAT DE L'ART

La section 2.1 introduit la planification d'itinéraires avec la perspective qu'un seul VÉ est présent. La section (2.2) introduit les autres VÉs mais sans pouvoir changer leur état (i.e. les planifier). Finalement, la section 2.3 aborde les planificateurs coopératifs tirés des travaux existants sur la planification coopérative multi-agent, entre autres motivée dans les jeux vidéos.

2.1 Planification d'un seul VÉ

Dans le cas d'un seul VÉ sur un réseau routier, quelques approches sont possibles.

2.1.1 Recherche dans un graphe de bornes

Une solution dans ce contexte est de d'abord créer un graphe des bornes tel que défini dans la section 1.2.3 de la formalisation. Celle-ci consiste à calculer les itinéraires les plus rapides entre toutes les bornes et remplacer l'itinéraire par une arête de coût équivalent. À partir du graphe des bornes, pour un seul VÉ dans un environnement statique, l'algorithme de Dijkstra ou l'algorithme A* peuvent être utilisés pour trouver l'itinéraire optimal.

Cette approche fonctionne bien avec les hypothèses listées au chapitre précédent mais, en pratique, elle a quelques limitations (Sachenbacher *et al.*, 2011).

2.1.2 Considération de l'élévation du terrain

Considérer la distance comme coût est limitant car la consommation de batterie est parfois négative si le VÉ est en pente descendante pour tout l'itinéraire (par exemple). Des facteurs inconnus avant le traitement de la requête (comme le poids du VÉ) influence l'autonomie, ce qui empêche un prétraitement calculant le coût entre les bornes. Finalement, la consommation de la batterie dépend de plusieurs

facteurs tel qu'expliqué dans la section 1.1.1 et le poids constants des arêtes du graphe des bornes ne traduisent pas cela.

(Sachenbacher *et al.*, 2011) aborde le problème en modifiant la génération du graphe des bornes et en modifiant A*. Pour le graphe des bornes, tous les itinéraires possibles entre les bornes sont calculés sans être limité par l'autonomie. Les auteurs ajoutent également l'information de hauteur pour chaque arête pour estimer l'influence du dénivelé sur la consommation de la batterie. Ensuite, ils traitent le fait que la batterie est vide ou pleine à l'aide de contraintes lors de la recherche de l'itinéraire. Les auteurs utilisent l'algorithme de Bellman-Ford.

2.1.3 Contraction de graphe

Sur un graphe de réseau routier, chaque appel à l'algorithme de Dijkstra ou même A* peut être coûteux pour de longs trajets dans de grandes cartes. Une technique de contraction hiérarchique de graphes permet de réduire significativement le temps de calcul (Geisberger *et al.*, 2012). un précalcul est cependant requis.

L'approche consiste à exploiter la hiérarchie inhérente de la structure d'un réseau routier (routes, autoroutes, etc.) pour créer de nouvelles arêtes créant des raccourcis lors de la recherche. Une hiérarchie de noeuds est créé : les noeuds les moins importants sont contractés avec des noeuds plus importants de manière récursive. Cette approche exacte et permet de trouver l'itinéraire optimal en quelques millisecondes sur un téléphone mobile pour une carte routière continentale (Eisner *et al.*, 2011).

2.2 Planification d'un VÉ en présence d'autres VÉs

Cette section fait état des approches pour planifier un seul VÉ, mais dans un environnement dynamique. Précisons que l'environnement est dynamique seulement à cause de la présence de d'autres VÉ aux bornes de chargement. Ainsi, on planifie un seul VÉ sans pouvoir contrôler les autres VÉ du réseau. Les impondérables de la conduite automobile (détours, incidents, temps d'attente à un arrêt ou à une lumière, etc.) créant une incertitude sur le temps de déplacement sont laissées de côté.

2.2.1 Modélisation et considération du temps d'attente aux bornes

Des modèles statistiques de comportements des usagers aux bornes de chargement existent (Lucas *et al.*, 2019) (Cao *et al.*, 2018).

Il est possible d'estimer les différentes actions que les VÉ effectuent aux bornes de chargement. Les auteurs de (Cao *et al.*, 2018) incluent le temps de stationnement (c'est-à-dire quand un VÉ occupe une borne sans se charger) ou encore les auteurs de(Franke et Krems, 2013) tentent de comprendre pourquoi les utilisateurs de VÉ n'empruntent pas l'itinéraire optimal et d'estime la surcharge du réseau des bornes en résultant. Ces modèles permettent l'estimation d'un temps d'attente à une borne donnée selon le moment d'arrivée d'un VÉ.

(Gareau *et al.*, 2019) estime le taux d'occupation des bornes selon l'heure en utilisant les données historiques pour ensuite intégrer des estimations des temps d'attentes dans les techniques de planification en environnement statiques discutés dans la section 2.1.

2.2.2 Planification avec contingence

Une autre manière de composer avec l'incertitude vis-à-vis la disponibilité des bornes est de retarder le moment de décider de s'engager à une borne. Cette approche est la planification avec contingence. Un exemple appliqué pour le calcul de chemin de drone piloté automatiquement est donné à la figure 2.1 (Hovenburg *et al.*, 2017).

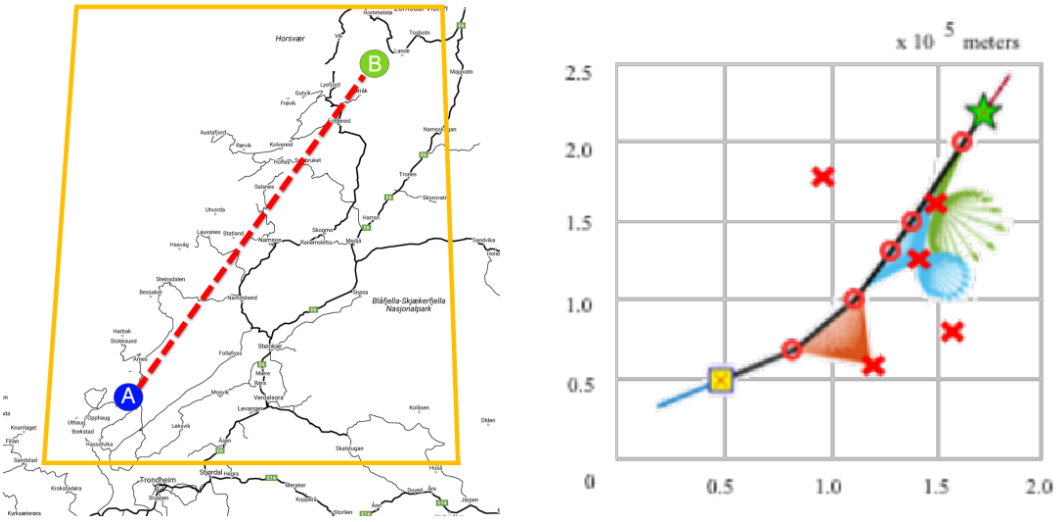


Figure 2.1 Un drone doit se rendre du point A au point B (image de gauche) et l'itinéraire optimal est une droite. En tenant compte du vent, des probabilités de bris et de l'autonomie, des zones d'atterrissage sont calculées (les X de l'image droite). Sont ensuite extraits des points de contrôles (les O de l'image de droite) par lesquelles le drone devra passer pour minimiser les risque de le perdre. Le compromis risque/longueur de l'itinéraire est paramétrable.

(Gareau *et al.*, 2019) ont proposée une méthode de planification contingence pour VÉ. Soient deux bornes candidates potentielles pour la recharge d'un VÉ. L'idée est d'emprunter un trajet mitoyen pour se rendre entre les deux bornes au lieu de se rendre directement à l'une d'elles. Trouver cet itinéraire est non trivial

en général. On s'engage à l'une des bornes en se dirigeant vers l'une d'elle à un moment lorsque l'incertitude sur la disponibilité est suffisamment faible. Trouver le bon moment pour s'engager est également non trivial. Les auteur de (Gareau *et al.*, 2019) utilisent une telle technique et ont développés une heuristique non-optimale pour trouver un moment pour s'engager.

2.3 Planification coopérative d'itinéraires

2.3.1 Recherche d'itinéraire multi-agent

Le problème peut également se résoudre dans le cadre de la recherche d'itinéraire multi-agent (Lejeune et Sarkar, 2021) et (Stern, 2019). Un problème type sur une grille est illustré sur la figure 2.2. Cette branche de la planification s'intéresse à la coordination d'agents qui doivent chacun aller d'un point de départ à un point d'arrivée. Ces approches sont utiles, par exemple, dans l'automatisation de la manipulation de marchandises dans les entrepôts par des robots, la livraison automatisée par des drones et la coordination de personnages non-joueur dans les jeux vidéos.

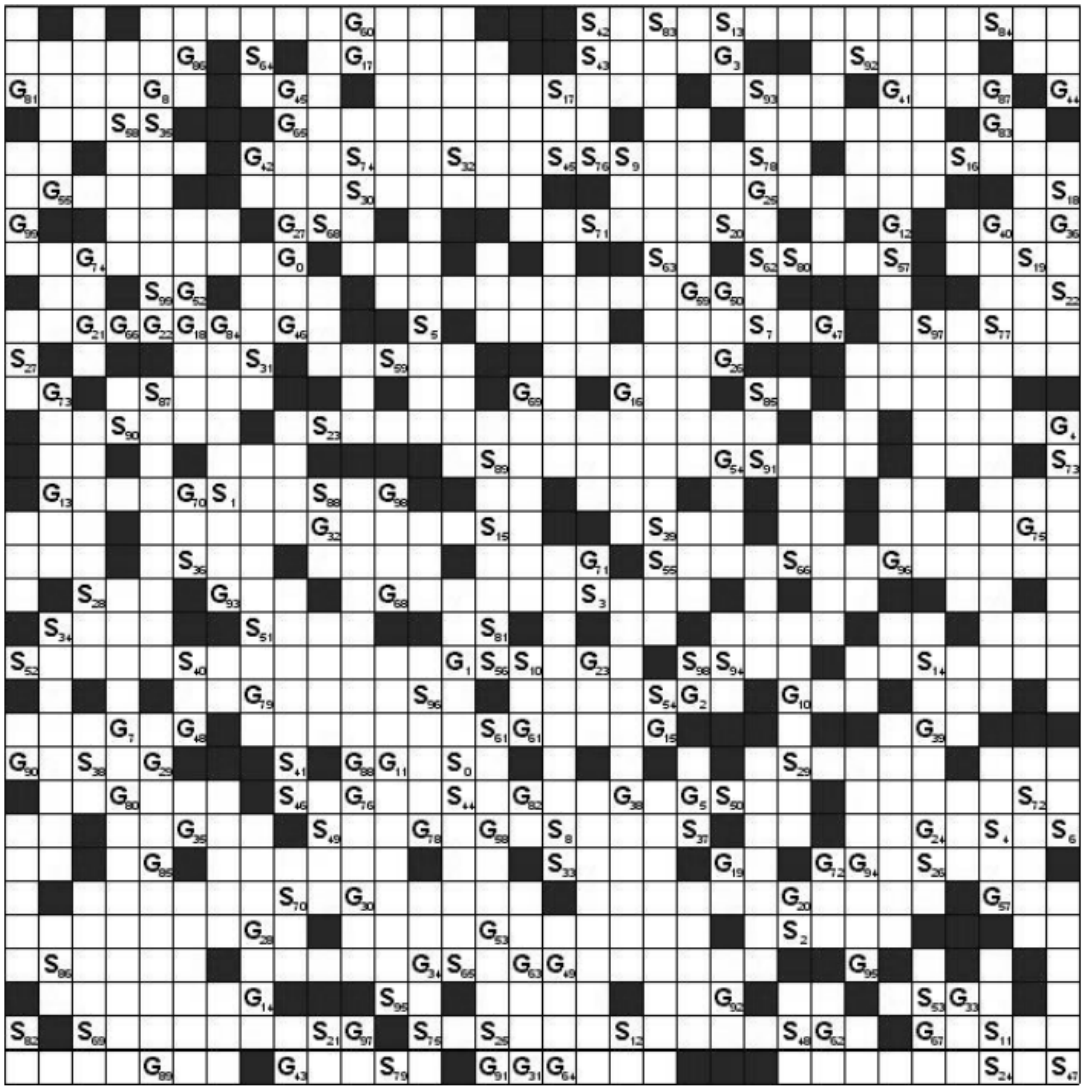


Figure 2.2 Exemple d'un problème canonique de planification d'itinéraire multi-agent. 100 agents se rendent de S_i à G_i. Les collisions entre les agents sont problématiques et trouver la solution optimale est non-trivial. Source : (Latombe, 1991).

2.3.2 Réparation locale pendant l'exécution

Une approche consiste à trouver un itinéraire pour chaque des agents (par exemple en utilisant A*) et lancer l'exécution. Pendant l'exécution, si une collision est imminente, on modifie l'itinéraire d'un/des deux agents : c'est la réparation locale pendant l'exécution. Par exemple, LRA* (*Local Repair A**) est une technique courante dans l'industrie du jeu vidéo (Hart *et al.*, 1968). Chaque agent utilise A* en ignorant les autres pour trouver l'itinéraire optimal et débutent l'exécution du plan. Lorsqu'une collision est éminente (un agent est trop proche/va en croiser un autre), un des agents reçoit l'ordre de faire un détour et recalcul son chemin à partir de sa position courante. Cette solution génère toutefois des plans invalides, nécessitant ainsi régulièrement une replanification, et est non-optimale.

2.3.3 Planification avec priorité

(Latombe, 1991) propose de résoudre ce problème en introduisant la planification A* coopérative. Cette technique consiste à trouver l'itinéraire optimal de chaque agent successivement comme pour la réparation locale pendant l'exécution, mais on ne lance pas l'exécution immédiatement. Pour chaque agent, l'itinéraire s'inscrit dans une table de réservation qui indique sa position à un temps donné. Les agents subséquent pourront ainsi planifier leurs itinéraires en contournant la position des autres agents en consultant la table de réservation. Dans cette optique, les agents ajoutés en premier ont la priorité sur les agents ajoutés en dernier. La figure 2.3 donne un exemple simple avec deux agents.

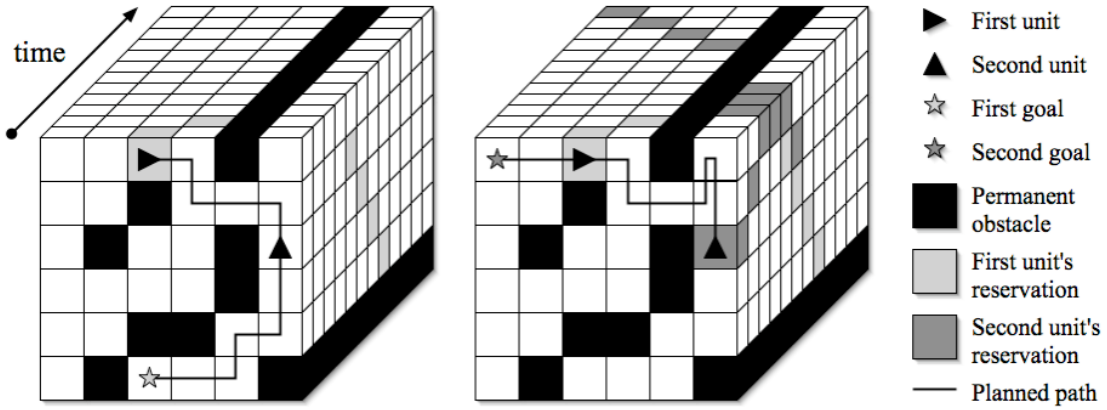


Figure 2.3 Exemple simple de l'algorithme A* coopératif. Le premier agent calcul son itinéraire optimal en inscrivant sa position dans la table de réservation. La profondeur représente le temps. Le second agent pourra trouver son trajet le plus cours en contournant le premier agent. Le premier agent a priorité sur le second. Source : (Silver, 2005).

Une contribution de (Latombe, 1991) est de comparer les solutions de A* coopératif en changeant l'ordre de calcul des agents dans le planificateur. Les ordres de calcul typiques sont :

1. Ordre d'insertion des agents dans le planificateur (une seule possibilité).
2. Toutes les permutations possibles (N! possibilités)
3. Ordre d'insertion cascadé (tableau 2.1) ($\frac{(N+1)N}{2}$ possibilités).
4. Ordre au hasard (H possibilités, avec H une constante au choix)
5. Ordre optimisé qui éviter les ordres de calcul symétriques, i.e. qui donne les mêmes itinéraires (Latombe, 1991)

Commentaire	VÉs échangés	Résultat
Ordre initial	A B C D	A B C D
Cascade de A	A B C D	B A C D
Cascade de A	A B C D	B C A D
Cascade de A	A B C D	B C D A
Cascade de B	A B C D	A C B D
Cascade de B	A B C D	A D C B
Cascade de C	A B C D	A B D C

Tableau 2.1 Sept ordres de calcul possibles pour quatre agents insérés en cascade.

L'algorithme M* répond à l'une question posée par (Latombe, 1991) : est-il possible de réduire le nombre de combinaisons à tester (Wagner et Choset, 2013) ? L'article propose de diviser l'espace en plusieurs sous-espaces indépendants (i.e. dans lesquelles un minimum de collisions entre les agents sont possibles) pour réduire la complexité de la recherche. Un exemple simple est donné à la figure 2.4.

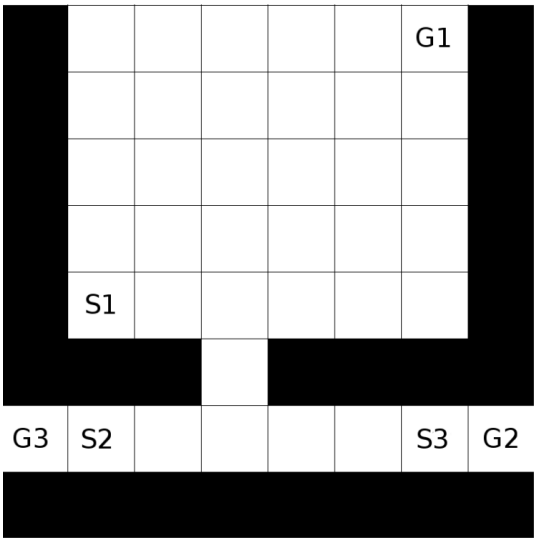


Figure 2.4 Exemple simple pour M*. Trois agents circulent de S_i à G_i. L'agent 1 ne croise aucunement les chemins des agents 2 et 3. L'agent 1 n'influence pas la planification des agents 2 et 3. Deux espaces de recherche sont créés : celui comprenant $\{\texttt{agent1}\}$ et l'autre, $\{\texttt{agent2}, \texttt{agent3}\}$. Le nombre de possibilités possible de calculs passe de $3! = 6$ à $1! + 2! = 3$. Tiré de (Wagner et Choset, 2013).

2.3.4 Planification avec priorité fenêtrée

Dans le but de rendre cette méthode exécutable en ligne (c'est-à-dire que le calcul peut s'effectuer pendant l'exécution du plan (Ghallab *et al.*, 2016)), (Silver, 2005) intègre les horizons temporels à l'exploration des ordres de calcul : seuls les conflits dans une horizon temporel fini sont tenues pour compte. Les agents ignorent complètement les conflits possibles après cette horizon temporel et utilisent A*. L'horizon temporel f, surnommée fenêtre, est ensuite avancée et le calcul reprend. Cette procédure a deux avantages : elle permet d'abord de trouver une solution arbitrairement rapidement si l'algorithme est interrompu. La solution peut cependant s'avérer invalide à l'exécution en cas de collision entre les agents.

Une petite fenêtre $\mathsf{f} \longrightarrow 0$ permet de trouver une meilleure solution mais allonge la durée de calcul. Une fenêtre $\mathsf{f} \longrightarrow \infty$ est identique à la planification avec priorité. Ainsi, la paramétrisation de la fenêtre temporelle donne un contrôle pour compromettre entre la qualité de la solution et la durée de calcul.

Un autre autre avantage est que, pendant l'exécution de la solution par les agents, la solution peut être raffinée en bougeant la fenêtre de temps en avant et en reprenant le calcul, évitant ainsi une replanification complète.

2.3.5 Collisions douces

On peut définir deux types de conflits pour des agents (Surynek, 2021) :

1. collision dure : il est impossible que deux agents partagent une même ressource ou la même position.
2. collision douce : des agents peuvent utiliser une même ressource même si cela détériore la solution.

Le problème abordé dans ce mémoire exhibe des propriétés de collision douce :

1. Plusieurs VÉ peuvent se trouver à une borne de chargement en même temps. Un seul VÉ peut utiliser la borne et les autres VÉ sont en attente, ce qui entraîne une détérioration de la solution.
2. Un VÉ peut passer par une borne de chargement sans être bloqué par un véhicule présent s'il n'a pas besoin de charger.

SC-M* spécialise l'algorithme M* dans le cas des collisions douces (Shi *et al.*, 2019) : advenant une collision à une position entre deux agents, on considère les itinéraires comme étant valides si le coût causé par la collision se trouve sous une valeur prédéfinie (ce seuil est paramétrable).

2.3.6 Approches par fusion de plans et résolution de conflits

Dans le contexte des collisions douces, CORALS utilise une technique différente (Benaskeur *et al.*, 2010). CORALS est un planificateur de défense naval qui coordonne des navires de combat de défense en réponse à une attaque. Pour créer une stratégie de défense efficace, un plan individuel optimal est créé pour chaque appareil. Par la suite, pour créer le plan global, les plans individuels sont successivement fusionnés puis altérés pour résoudre les conflits entre les actions.

CHAPITRE III

PLANIFICATEURS HORS LIGNE

Dans les environnements statiques, la planification peut être effectuée hors ligne, c'est-à-dire que des plans peuvent être trouvés et évalués avant l'exécution. Dans des environnements dynamiques, un plan doit être généré pendant l'exécution (section 2.6.2 de (Ghallab *et al.*, 2016)).

Dans ce chapitre, les précalculs sont d'abord détaillés. Des planificateurs hors ligne (coopératifs ou non) sont par la suite présentés. Les planificateurs en ligne sont l'objet du prochain chapitre.

3.1 Graphe des bornes

La distance des déplacements entre chaque paire de bornes est précalculée. L'application de l'algorithme A* (mentionné dans la section 2.1.1 de l'état de l'art) pour créer G_b est détaillée dans l'algorithme 1. Rappelons que, dans ce contexte, les itinéraires optimaux sont ceux pour lesquels un VÉ serait seul sur la route pour se rendre d'une borne à l'autre (figure 3.1). La présence du paramètre η_{max} limite le nombre d'arêtes dans G_b.

En effet, dans le graphe des bornes, chaque borne a des arêtes vers toutes les autres bornes situées à maximum de $\eta_{max} = 400$km. La majorité des VÉ sur le marché ont moins de 400 km d'autonomie, ce qui justifie la sélection de η_{max} dans ce mémoire, mais η_{max} reste un paramètre modifiable.

La nécessité de planifier les trajets devient moindre lorsque l'autonomie augmente. Le nombre d'arêtes est proportionnel au carré de ce paramètre. Si $\eta_{max} \longrightarrow \infty$, le nombre d'arêtes de G_b devient égal à $|B|^2$, c'est-à-dire que G_b devient un graphe complet. Augmenter η_{max} allonge également la durée de calcul.

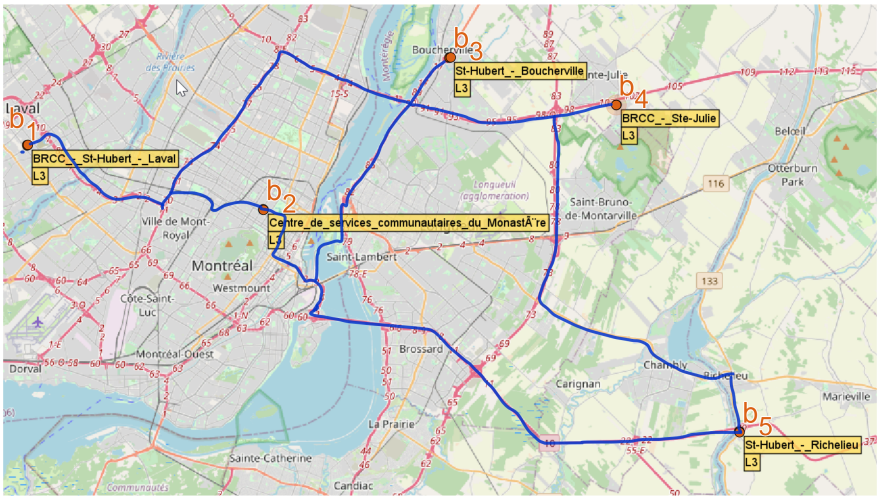


Figure 3.1 Itinéraires optimaux entre cinq bornes de la région de Montréal.

Les planificateurs globaux n'ont pas besoin de connaître le détail des plans locaux : ils planifient directement sur le graphe des bornes tel qu'illustré sur la figure 3.2. Autrement dit, seulement le temps de déplacement$W(\alpha)$ des arêtes $\alpha \in G_b$ est conservé. Les trajets, en termes de noeuds de la carte, sont recalculés après au besoin.

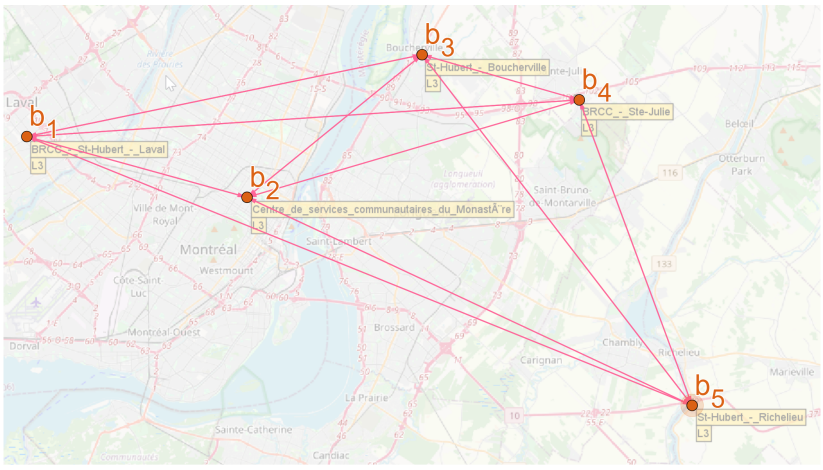


Figure 3.2 Exemple de graphe des bornes. Les détails des itinéraires entre les bornes sont substitués par une arête α ayant un poids $W(\alpha)$ correspondant à la distance de déplacement le plus cours.

Algorithme 1 : Calcul du graphe des bornes.

Entrée : Carte routière $G = (S, A)$
Liste de bornes B
Sortie : Graphe des bornes G_b

```
1  Fonction GenerationGrapheDesBornes(G, B) :
2      G_b ← ∅
3      pour Borne b ∈ B faire
4          G_b ← G_b ∪ {b}
5          pour Borne b' ∈ B/b faire
6              π_routier ← A*(b, b', G)
7              //L'arête α n'est pas créée si l'autonomie η_max dépasse M_durée(α)
8              si π_routier ≠ ∅ & ζ < M_durée(α) alors
9                  α ← (b, b')
10                 W(α) ← M_durée(α)
11                 G_b ← G_b ∪ {α}
12     retourner G_b
```

3.1.1 Graphe des bornes augmenté

Lorsqu'un VÉ s'insère sur la route, il ne quitte pas nécessairement directement d'un noeud attaché à une borne de recharge. Deux bornes virtuelles, une correspondant à l'origine et l'autre à la destination, sont temporairement ajoutées au graphe des bornes. On doit également les accompagner de leurs arêtes. On crée de cette manière le graphe des bornes augmenté. La figure 3.3 illustre l'insertion des deux bornes virtuelles et des itinéraires calculés. Sur la figure 3.4, on trouve le graphe des bornes augmenté correspondant. Dans le cas particulier que l'origine est une borne de recharge, il n'est pas nécessaire de supplémenter l'origine. Le raisonnement est symétrique pour un point d'arrivée.

Aucune recharge n'est possible pour les deux bornes virtuelles, ce qui cadre correctement avec l'hypothèse sur l'état de la batterie au départ et l'arrivée (section 1.1.1 des hypothèses).

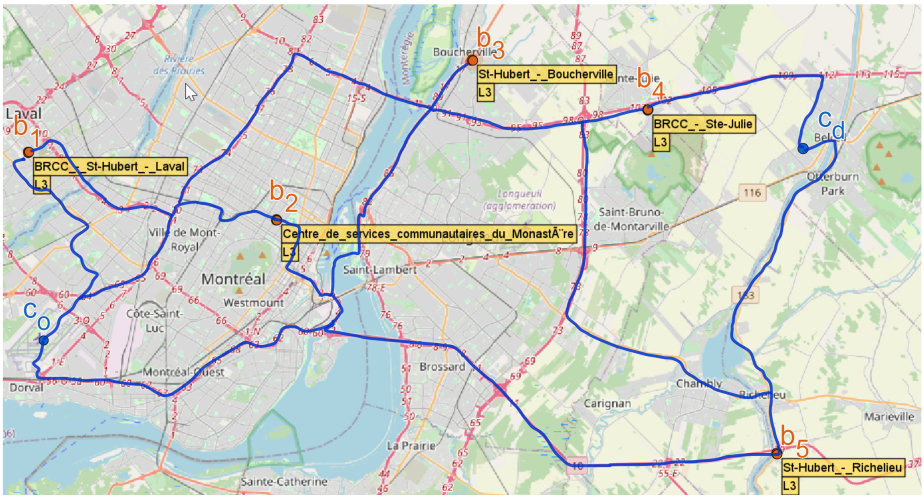


Figure 3.3 Chemins les plus courts entre des bornes, l'origine c_o et la destination c_d.

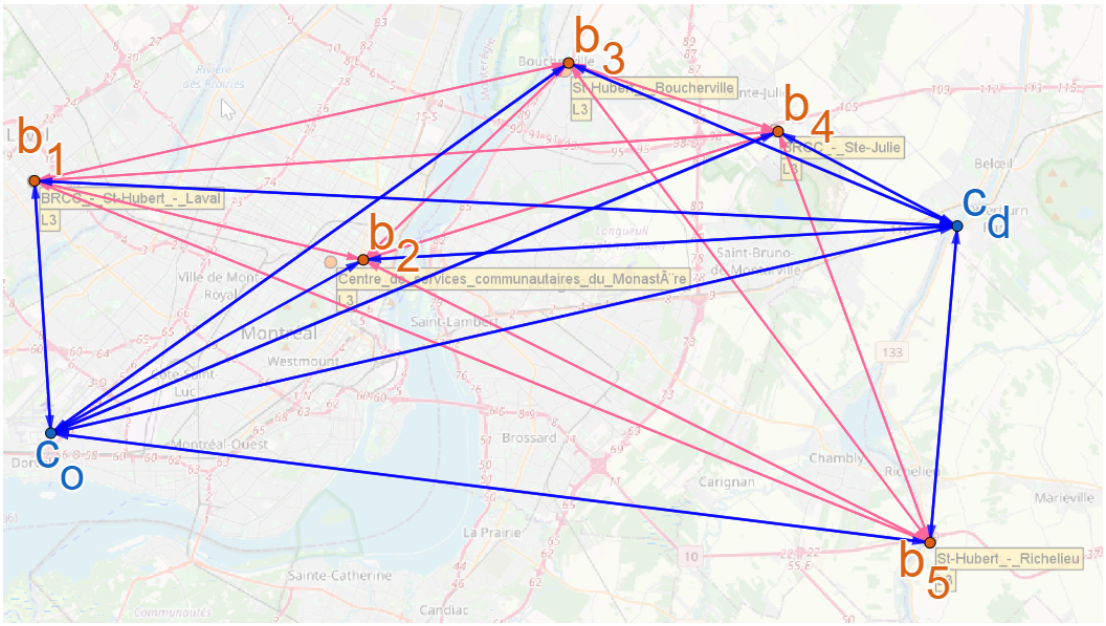


Figure 3.4 Graphe des bornes augmenté pour un VÉ.

Il est important de constater que pour chaque VÉ correspond un graphe des bornes augmenté, mais le même graphe des bornes est simplement supplémenté, et ce, sans être copié. Pour chaque VÉ, les suppléments sont précalculés pendant l'exécution avant que les planificateurs ne débutent leur exécution. Dans les explications subséquentes, ce précalcul ne sera pas mentionné même si omniprésent. Il affecte cependant la performance et sera discuté dans la section des résultats.

Pour calculer les arêtes associées aux bornes virtuelles, l'algorithme du calcul du graphe des bornes 1 est utilisé, mais la liste de bornes B est remplacée par $B' = \{c_o, c_d\}$. Les planificateurs travaillent dans l'espace du graphe des bornes augmenté et non sur la carte routière. Ce faisant, le nombre de noeuds sur lesquels calculer est $|B| \ll |S|$: le nombre de bornes de chargement rapide au Québec est de 463 (Le Circuit Électrique, 2021) mais 4.2 millions de noeuds sont présents sur de la zone du Québec de la carte routière d'OpenStreetMap.

3.2 Planification hors ligne

Une planification hors ligne signifie qu'un planificateur global ont accès à toutes les requêtes du système, et ce, même si les temps de départ des requêtes sont différents (Ghallab *et al.*, 2016). Dans la réalité, de nouveaux véhicules s'insèrent constamment sur le réseau routier et seulement les requêtes effectuées jusqu'à ce moment sont accessibles. Autrement dit, ce sont des départs en continu et des planificateurs dits en ligne sont préférables. Ce type de méthodes sera étudié dans le chapitre 4 des planificateurs en ligne.

3.3 Méthode PI : Planification individuelle

Le *planificateur individuel* ne considère aucune coopération entre les VÉ et ignore les conflits aux bornes : il n'y a donc pas d'attente. Inspiré de LRA*, c'est l'équivalent de calculer l'itinéraire optimal pour chaque requête $\mathtt{r} \in \mathtt{R}$. Ceci est détaillé à l'algorithme 2 (Hart *et al.*, 1968).

Autrement dit, les actions possibles des VÉ sont uniquement :

1. avancer à la prochaine borne
2. charger à une borne

Algorithme 2 : Calcul de PI .

Entrée : $\mathtt{G_b}$: Graphe des bornes.
$\mathtt{R} = \langle \mathtt{r}_1, \mathtt{r}_2, ..., \mathtt{r}_{|\mathtt{R}|} \rangle$: suite de requêtes.
Sortie : π : Plan de la flotte de VÉ.

```
1 Fonction PI(G_b, R) :
2     π ← ∅
3     pour r = (c_o, c_d, τ) ∈ R faire
4         π_r ← A*(c_o, c_d, G_b)
5         π ← π ∪ {π_r}
6     retourner π
```

Le coût du plan global π est calculé avec l'équation (1.6) de la section de formalisation, mais $\mathtt{d}_{\mathtt{attente}}$ étant toujours $\mathtt{0}$. Le coût du plan global généré par PI fait office de borne inférieure (corresponds à l'équation (1.9)) et permet le calcul de la pénalité décrit à la section 1.10.

3.4 Système de réservation aux bornes

Contrairement au problème classique de planification de chemin multiagent, deux VÉ peuvent se trouver à une même borne. L'un d'eux peut simplement attendre que la borne se libère pour ensuite l'occuper, et ce, sans empêcher la circulation. C'est la notion de *collision douce* abordée dans la section 2.3.5.

3.4.1 Réservation et agenda

Il existe un agenda pour chaque borne. Une réservation est un triplet $\rho = (\mathfrak{b}, \tau_{\text{arrivee}}, \tau_{\text{départ}})$. L'agenda d'une borne $\mathfrak{b}$ est une liste de réservations est $R_{\mathfrak{b}}$. Les intervalles de temps de $\rho \in R_{\mathfrak{b}}$ ne peuvent pas avoir d'intersections entre eux. L'agenda global Γ est une application de $\mathsf{B} \rightarrow R_{\mathfrak{b}}$ qui donne l'agenda d'une borne.

Le fonctionnement d'une borne est donc comme suit : lorsqu'un VÉ se charge, une réservation ρ est ajoutée dans son agenda $R_{\mathfrak{b}}$. Deux VÉ ne peuvent pas avoir de réservations concurrentes : au mieux, un des VÉ devra attendre son tour à la manière d'une file d'attente le temps que la borne se libère. Aucune recharge partielle n'est permise (hypothèse 1.1.1). Il n'est également pas possible de déplacer une réservation existante. Autrement dit, c'est premier réservé, premier servi.

3.4.2 Chronométrage

Le fonctionnement d'une borne est donc comme suit : lorsqu'un VÉ se charge, une réservation ρ est ajoutée dans son agenda $R_{\mathfrak{b}}$. Les réservations concurrentes ne sont pas possibles : au mieux, un des VÉ devra attendre son tour à la manière d'une file d'attente le temps que la borne se libère. Aucune recharge partielle n'est permise (hypothèse 1.1.1). Il n'est également pas possible de déplacer une réservation existante.

Algorithme 3 : Chronométrage du déplacement, de l'attente et du chargement lors du parcours d'une arête $\alpha = (b_o, b_d)$.

. Entrée : τ_o : Heure de départ de b_o
$\alpha = (b_o, b_d)$: arête parcourue
Sortie : τ_d : Heure de départ de b_d
Fonction `Chronometrage`(τ_o, α) :
    //Selon l'équation (1.3)
    $d_{deplacement} \leftarrow \frac{W(\alpha)}{v}$
    $\tau_{arrive} = \tau_o + d_{deplacement}$
    //Selon le calcul 1.2.5
    $d_{chargement} \leftarrow \psi \times d_{deplacement}$
    $\tau_{disponible} \leftarrow \mathsf{TrouverProchainePlage}(\Gamma(b_d), \tau_{arrive}, d_{chargement})$
    $d_{attente} \leftarrow \tau_{disponible} - \tau_{arrive}$
    $\tau_d \leftarrow \tau_{arrive} + d_{attente} + d_{chargement}$
    retourner τ_d

3.4.3 Effectuer une réservation

L'algorithme 4 expose la procédure pour effectuer les réservations à partir d'un itinéraire. Cet algorithme fait usage de l'algorithme 5 qui démontre comment réserver une borne lors du parcours d'une arête α.

Algorithme 4 : À partir d'un itinéraire, les réservations peuvent être effectuées en parcourant chaque arête α parcourue.

Entrée : π_{v_i} : Itinéraire de v_i
Γ : Agenda global de G_b.
Fonction `ReserverAgenda`(π_{v_i}, Γ) :
    pour $\alpha = (b_o, b_d) \in \pi_{v_i}$ faire
        `ReserverBorne`(α, Γ)

La structure accueillant les réservations est un arbre de recherche binaire. L'insertion est plus lente que si l'on utilisait une liste, mais la consultation est rapide pour la fonction $\mathsf{TrouverProchainePlage}$. C'est avantageux, car les planificateurs effectuent davantage de vérification de disponibilité que de réservations.

Algorithme 5 : Procédure de réservation à une borne.

Entrée : $\alpha = b, b'$: Arête liant une borne b à une borne b'.
Γ : Agenda global de G_b.

1 Fonction `ReserverBorne`$(\alpha = (b, b_d), \Gamma)$:
2 $\quad \tau_{disponible} \leftarrow \mathsf{TrouverProchainePlage}(\Gamma(b_d), \tau_{arrive}, d_{chargement})$
3 $\quad d_{attente} \leftarrow \tau_{disponible} - \tau_{arrive}$
4 $\quad \tau_{départ} \leftarrow \tau_{disponible} + d_{chargement}$
5 $\quad \mathsf{AjouterReservation}(\Gamma(b_d), \tau_{disponible}, \tau_{départ})$

3.4.4 Algorithme A* adapté à la planification d'un VÉ

Afin d'accommoder les particularités de la planification de VÉ, une version modifiée de l'algorithme A* est présentée à l'algorithme 6. Il y a trois différences importantes par rapport à l'original. La première concerne la seconde condition de la ligne 15 : la distance de déplacement $W(\alpha)$ sur l'arête α doit être inférieure à l'autonomie du VÉ γ sinon l'arête est ignorée. La seconde différence est à la ligne 16 : le coût est donné par la fonction de `Chronometrage` (algorithme 1) et non uniquement $W(\alpha)$. La troisième différence est à la ligne 20 : l'heuristique doit s'exprimer en durée et non en distance. La distance euclidienne peut être transformée en durée en divisant par la vitesse υ en se référant à l'équation (1.3). Il aurait été possible de rendre l'heuristique plus informative en incorporant la durée de chargement $d_{chargement}$ étant donné qu'on a simplement besoin de multiplier par ψ pour l'obtenir, mais l'heuristique ne serait plus admissible, car la dernière borne n'a pas besoin de charger (hypothèse 1.1.1).

Algorithme 6 : A* modifié avec l'autonomie maximum γ du VÉ et le chronométrage.

Entrée : G_b : Graphe des bornes.
$r = (b_o, b_d, \tau)$: Une unique requête.
Sortie : π_r : Itinéraire associé à la requête r.

```
1  Fonction A*Modifié(G_b = (B, A), r = (b_o, b_d, τ)) :
2      pour b ∈ B faire
3          durees(b) ← ∞
4          dureesEstimees(b) ← ∞
5          parents(b) ← ∅
6          visites(b) ← false
7      durees(b_d) ← τ
8      dureesEstimees(b_d) ← τ
9      visites(b_d) ← true
10     tant que true faire
11         b' ← NoeudAvecLaPlusPetiteDuree()
12         visites(b') ← true
13         pour arête sortante α = (b', c) depuis le sommet b' faire
14             //Une borne doit être accessible par le VÉ pour être visité.
15             si ¬visites(c)etW(α) < γ alors
16                 chronometrage_b ←Chronometrage(durees(b), α)
17                 nouvelleDuree ← durees(b) + chronometrage_b
18                 si nouvelleDuree < durees(c) alors
19                     durees(c) ← nouvelleDuree
20                     dureesEstimees(c) ←
                        nouvelleDuree + DistanceEuclidienne(α) / υ
21                     parents(c) ← b'
22             si b' = b_d alors
23                 retourner ConstruireChemin(b_d)
24             si durees(NoeudAvecLaPlusPetiteDuree()) → ∞ alors
25                 retourner ∅
```

3.5 Méthode PNC : Planification non coopérative

L'idée de PNC est analogue à PI mais il y a une résolution des conflits. Les VÉ ne savent pas d'avance si des VÉ seront présents aux bornes et emprunteront le même trajet que PI . Un conflit de réservation a une borne va générer une attente pour les VÉ subséquents la réservant.

Le plan proposé par PNC correspond à la situation dans laquelle chaque VÉ emprunte l'itinéraire individuel optimal, mais pourrait devoir attendre à une borne si celle-ci est déjà réservée par un autre VÉ. Cette solution constitue une borne supérieure. L'algorithme 7 détaille la procédure.

Algorithme 7 : Calcul de PNC.

Entrée : $\mathsf{G_b}$: Graphe des bornes.
$\mathsf{R} = \langle \mathsf{r}_1, \mathsf{r}_2, ..., \mathsf{r}_{|\mathsf{R}|} \rangle$: suite de requêtes.
Sortie : π : Plan de la flotte de VÉ.

```
1 Fonction PNC(G_b, R) :
2     π ← ∅
3     pour r = (c_o, c_d) ∈ R faire
4         π_r ← A*Modifié(c_o, c_d, G_b)
5         π ← π ∪ {π_r}
6     pour π_r ∈ π faire
7         ReserverAgenda(π_r, Γ)
8     retourner π
```

3.6 Méthode PNCAR : Planification non coopérative avec réservations

L'algorithme 8 présente le planificateur PNCAR. La principale différence avec la méthode PNC est qu'elle intègre un planificateur individuel qui prend en compte les réservations des autres VÉ aux bornes.

Chaque requête est planifiée successivement dans l'ordre de la suite de requêtes

(ligne 3 de l'algorithme 8). Après avoir planifié l'itinéraire d'une requête, l'algorithme réserve la(les) borne(s) dans l'agenda Γ. Ces réservations empêchent que les requêtes suivantes utilisent les bornes en même temps. De cette manière, pour chaque requête, on trouve l'itinéraire individuel optimal en considérant les requêtes passées.

Avec PNCAR, les VÉs ne coopèrent pas : ils cherchent à minimiser individuellement leur itinéraire respectif en fonction des requêtes précédentes. Une observation est que le premier VÉ de la requête n'aura jamais d'attente (l'agenda est garanti d'être vide) et empruntera l'itinéraire optimal donné par A*. À l'inverse, le dernier VÉ planifié est celui qui, en général, pourrait devoir attendre ou contourner tous les autres VÉ. Ainsi, comme discuté dans l'état de l'art dans la section 2.3.3, l'ordre dans lequel les VÉ calculent leur trajet influence la qualité du plan global.

Algorithme 8 : Calcul de PNCAR.

Entrée : G_b : Graphe des bornes.
$R = \langle r_1, r_2, ..., r_{|R|} \rangle$: suite de requêtes.
Sortie : π : Plan de la flotte de VÉ.

```
1 Fonction PNCAR(G_b, R) :
2     π ← ∅
3     pour r = (c_o, c_d) ∈ R faire
4         π_r ← A*Modifié(c_o, c_d, G_b)
5         π ← π ∪ {π_r}
6         ReserverAgenda(π_r, Γ)
7     retourner π
```

3.7 Méthode PCC : Planification coopérative complète

Une façon de garantir un plan optimal est une exploration exhaustive de tout l'espace des plans Π (Land et Doig, 1960). L'algorithme 9 calcule, pour chaque requête, tous les itinéraires possibles, et ce, en explorant tous les ordres d'arrivée possibles aux bornes. L'énumération de ces itinéraires est effectuée avec une recherche en profondeur récursive. Le principe est illustré sur la figure 3.5. On remarque que la ligne 17 de l'algorithme utilise une des six fonctions objectives de la section 1.2.5 pour déterminer le meilleur plan global.

La profondeur maximale de l'arbre de la figure 3.5 est $|\mathsf{R}| \times |\mathsf{B}|$. Le facteur de branchement à chaque niveau est variable et dépend de l'autonomie de chaque VÉ, mais dans cette situation, on peut exprimer la complexité comme étant $\Omega(2^{|\mathsf{R}|\times|\mathsf{B}|})$. Cela rend PCC impraticable : le terme $|\mathsf{R}|\times|\mathsf{B}|$ est généralement élevé en pratique. Par exemple, pour 10 voitures sur un réseau de 500 bornes, on au minimum 2^{5000} états à explorer.

En terme de complexité de mémoire, étant une recherche en profondeur, deux plans globaux sont comparés et un seul est gardé. La profondeur maximum est donc toujours $2 \times |\mathsf{R}| \times |\mathsf{B}|$, ce qui correspond à une complexité $\Omega(|\mathsf{R}| \times |\mathsf{B}|)$.

Algorithme 9 : Exploration complète de l'espace de plan Π avec PCC .

Entrée : G_b : Graphe des bornes.
$R = \langle r_1, r_2, ..., r_{|R|} \rangle$: suite de requêtes.
π : Plan de la flotte de VÉ.
Γ : Agenda global de G_b.

Sortie : π : Plan de la flotte de VÉ.

```
1  Fonction PCC(G_b, R, π, Γ) :
2      si R = ∅ alors
3          retourner π
4      π_meilleur ← ∅
5      M(π_meilleur) ← ∞ pour r = (c_o, c_d) ∈ R faire
6          pour arête sortante α = (b_i, b_d) depuis le sommet b_i faire
7              //Une borne doit être accessible par le VÉ pour être visité.
8              si b_d ∈ π_r alors
9                  retourner ECHEC
10             π_r.ajouter(b_d)
11             ReserverBorne(α, Γ)
12             si b = c_d alors
13                 r' ← (b_d, c_d, τ_départ(b_d))
14                 //Substitue la requête r par r' dans la suite R.
15                 R.remplacer(r, r')
16             π_candidat ←PCC(G_b, R, π, Γ)
17             si M(π_candidat) < M(π_meilleur) alors
18                 π_meilleur ← π_candidat
19     retourner π_meilleur
```

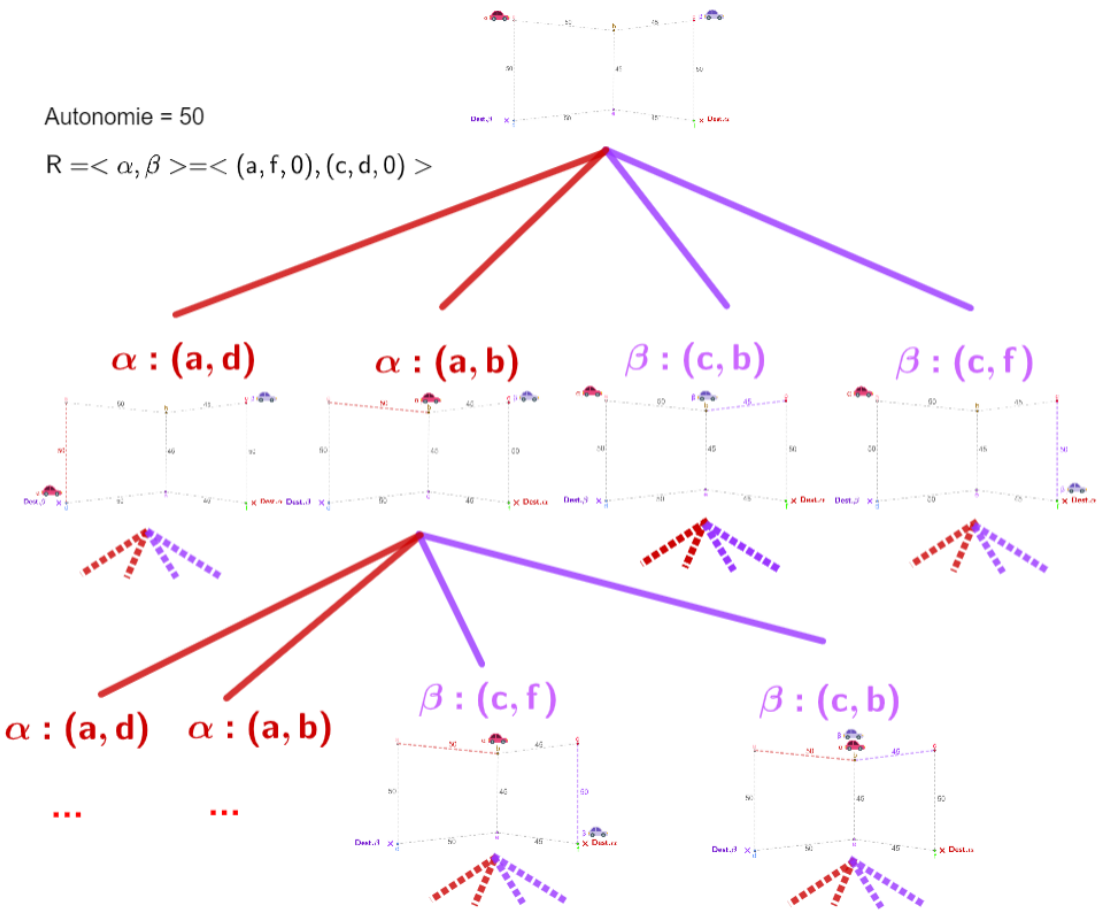
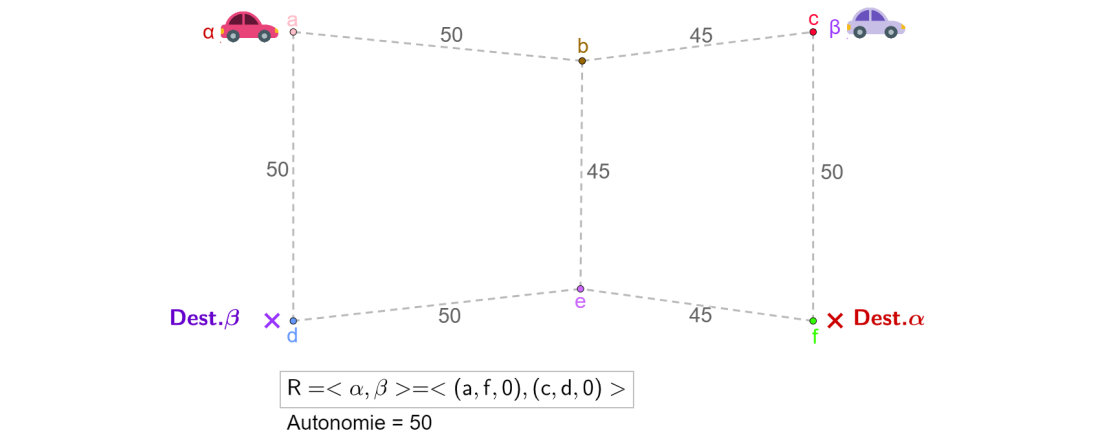


Figure 3.5 Illustration de la planification complète à partir de l'exemple 3.6. Lors du parcours d'une arête par un VÉ, toutes les autres possibilités de parcours pour l'ensemble de la flotte sont explorées.

3.8 Mise en situation

Appuyés par un exemple, des plans issus de PI, PNC, PNCAR et PCC sont illustrés sur la figure 3.6. La fonction objectif de durée des itinéraires 1.6 est utilisée. Les conflits sont mis en évidence en rouge hachuré. On remarque que PNC donne le meilleur plan Π, mais celui-ci est invalide, car il n'y a pas de gestion de conflits. Le meilleur plan avec gestion de conflits est celui créé par PCC.



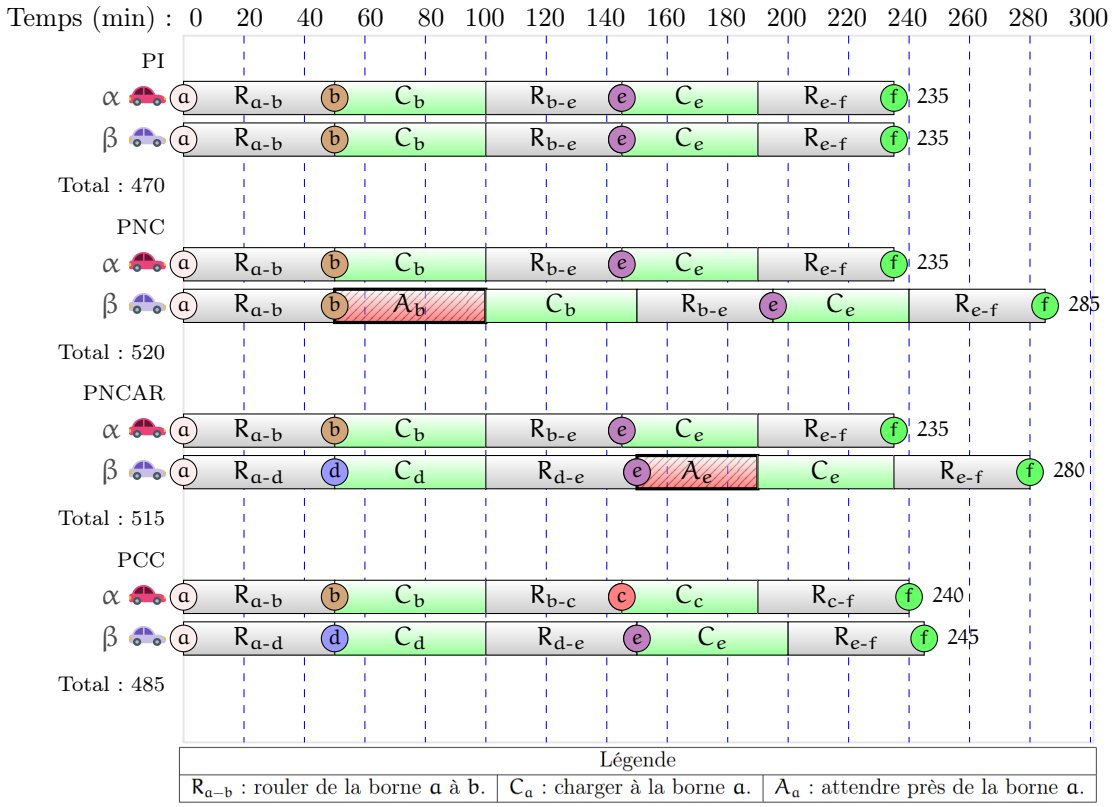


Figure 3.6 Comparaison des plans générés par PI, PNC, PNCAR et PCC.

3.9 Méthode PC-P : Planification coopérative - permutation

PC-P est une extension de PNCAR : tel que mentionné pour PNCAR, l'ordre dans laquelle les VÉ calculent leur itinéraire influence le plan global généré. L'idée centrale de PC-P est de comparer toutes les permutations possibles de R, puis de calculer PNCAR. On sélectionne le meilleur plan parmi ceux proposés. Cette approche est initialement proposée dans (Latombe, 1991) dans un contexte de collision dure sans file d'attente. Cette idée a été discutée dans la section 2.3.3.

L'algorithme 10 détaille la procédure. La ligne 5 donne toutes les permutations possibles de $\mathsf{r} \in \mathsf{R}$. Ce planificateur ne garantit pas de trouver un plan global optimal. La figure 3.7 présente une mise en situation de l'application de PC-P avec 3 VÉ. $3! = 6$ ordres de calcul sont donc possibles. Certains ordres de calcul sont équivalents et mènent au même plan, comme les 2 dernières. Les meilleurs plans sont le premier, le troisième et le quatrième qui ont tous la même valeur de fonction objective de 475. Remarquons que les plans ne sont pas tous identiques.

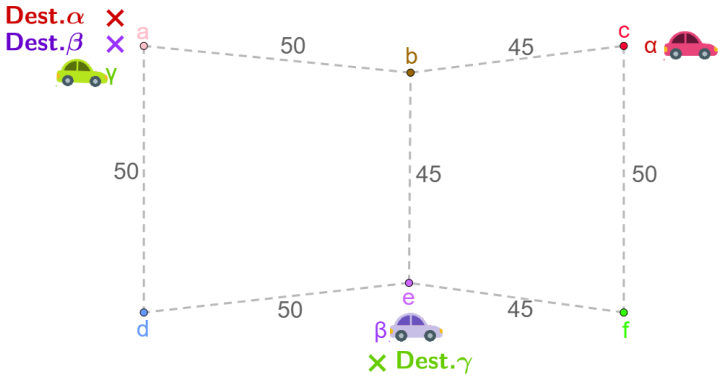
Algorithme 10 : Calcul de PCNP.

Entrée : $\mathsf{G_b}$: Graphe des bornes.
$\mathsf{R} = \langle \mathsf{r}_1, \mathsf{r}_2, ..., \mathsf{r}_{|\mathsf{R}|} \rangle$: suite de requêtes.
Sortie : π : Plan de la flotte de VÉ.

```
1  Fonction PCNI(G_b, R) :
2      π ← ∅
3      M(π) ← ∞
4      //Le résultat de la ligne suivante dépend des ordres de calcul explorés
         (Permutation ou cascade).
5      P ← CréerPermutationsRequêtes(R)
6      pour R' ∈ P faire
7          π_candidat ← PNCAR(G_b, R')
8          si M(π_candidat) < M(π) alors
9              π ← π_candidat
```

L'algorithme PC-P considère toutes les permutations possibles et est par consé-

quent $|V|!$ plus complexe que PNCAR. L'avantage de cette méthode est la réduction de l'exploration de l'espace de plans Π par rapport à PCC .



$R =< \alpha, \beta, \gamma >=< (c, a, 0), (e, a, 0), (a, e, 0) >$

Autonomie = 50

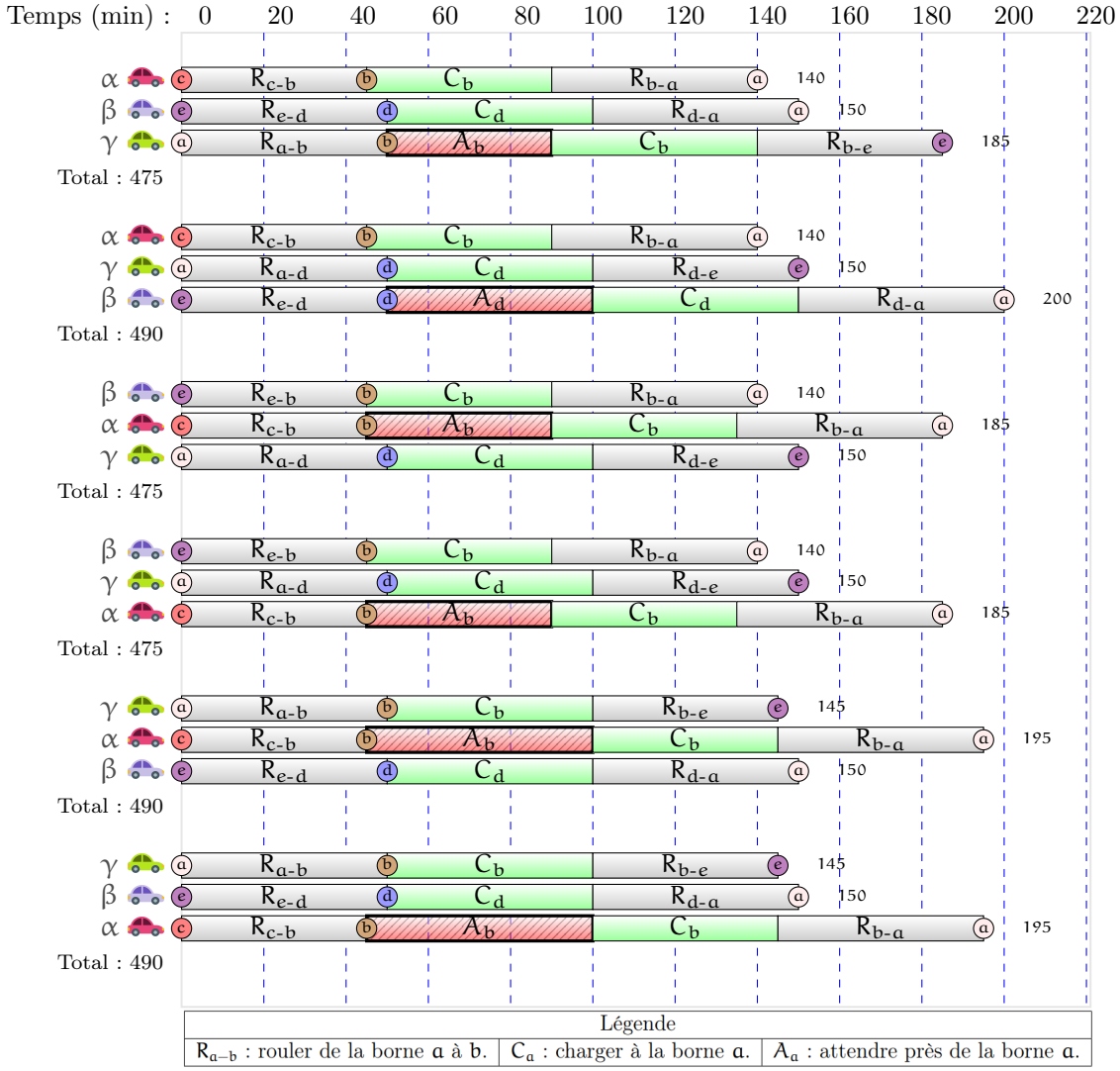


Figure 3.7 Exemple de mise en application de PC-P avec 3 VÉ.

3.10 Méthode PC-C : Planification coopérative - cascade

Tel que mentionné dans 2.3.3, d'autres ordres de calcul sont possibles (Latombe, 1991). PC-C utilise la méthode d'ordre en cascade. Rappelons que cette méthode est détaillée dans le tableau 2.1. Elle est $|V|^2$ plus complexe que PNCAR, ce qui est significativement préférable à $|V|!$ de PC-P de la sous-section précédente.

On peut adapter l'algorithme 10 en remplaçant la ligne 5 pour créer les ordres de calcul en cascade. La figure 3.8 présente une mise en situation de l'application de PC-C avec 3 VÉ. $(3-1)^2 = 4$ ordres de calcul sont possibles. Les trois premiers plans sont les meilleurs et identique en termes de valeur de fonction objective.

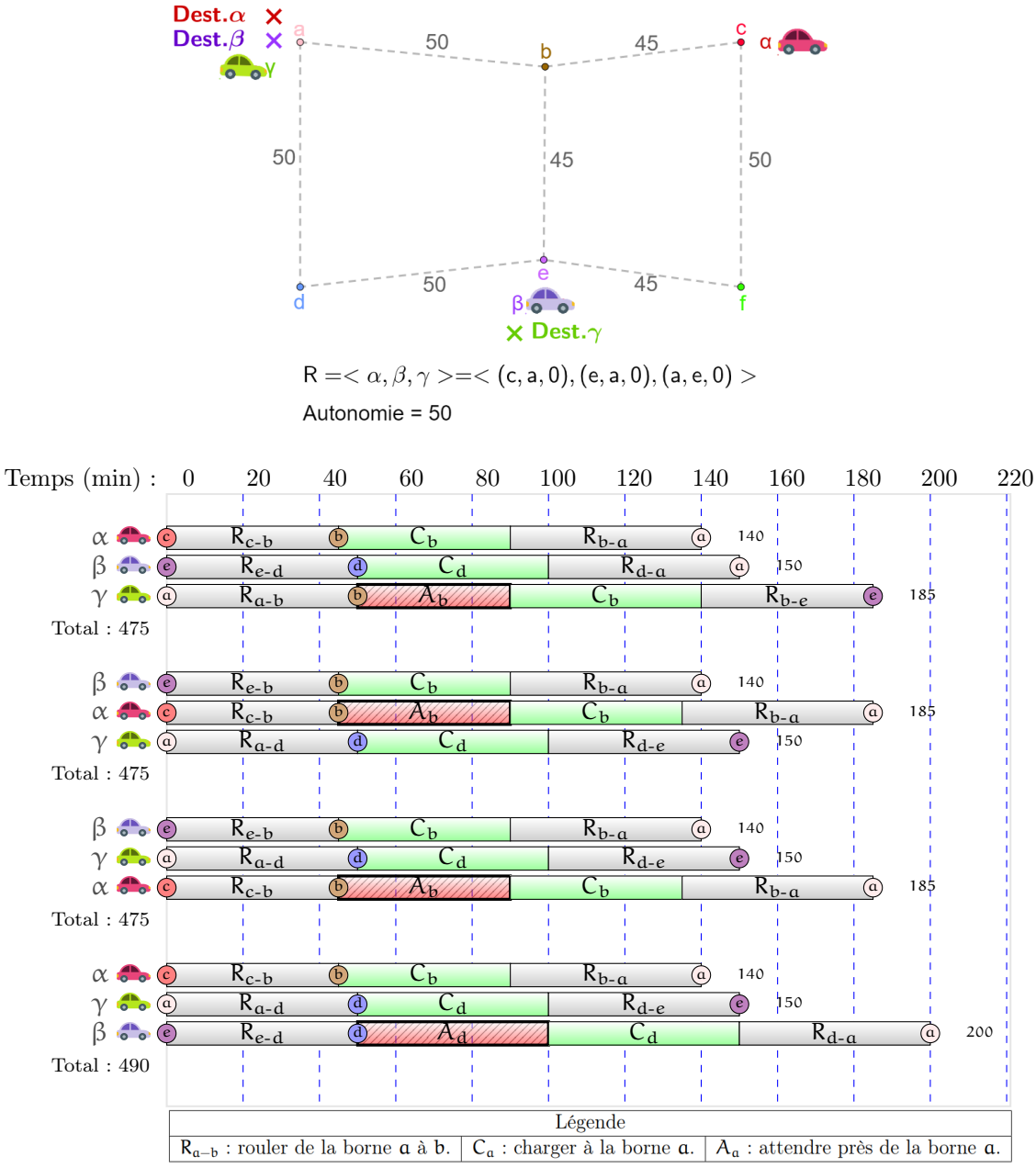


Figure 3.8 Exemple de mise en application de PC-C avec 3 VÉ.

CHAPITRE IV

PLANIFICATEURS EN LIGNE

Les planificateurs du précédent chapitre sont conçus pour travailler hors ligne : l'ensemble de requêtes est accessible par les planificateurs en tout temps. Dans la réalité, des VÉ s'insèrent continuellement. De plus, les VÉ rendus à destination sont retirés : R est donc dynamique. Étant donné que R change continuellement et que la planification n'est pas instantanée, les planificateurs doivent s'exécuter pendant la réalisation des plans : c'est la planification en ligne.

4.1 Replanification

Une méthode connue pour utiliser dans un contexte en ligne un planificateur hors ligne est la replanification tel que discuté dans le chapitre 3 de (Norvig Peter et Russell, 2020). Cela revient à utiliser les planificateurs hors ligne de la section 3.2 périodiquement. L'état du système au moment choisi est mis en entrée dans le planificateur.

La sélection du moment de la replanification n'est pas triviale et plusieurs options sont possibles. Par exemple, on peut replanifier à chaque ajout de requêtes dans R. Cette méthode est inefficace si plusieurs VÉ s'insèrent sur la route pendant que le planificateur s'exécute déjà : soit la planification en cours est suspendue, ou encore elle est relancée seulement quand la planification précédente a terminée.

Une méthode alternative est d'accumuler les nouvelles requêtes dans une liste temporaire et, à intervalle régulier, exécuter le planificateur en ajoutant toutes les nouvelles requêtes de la liste temporaire. Le planificateur ne sera ainsi pas interrompu si un VÉ s'insère sur la route, mais un VÉ peut devoir attendre plus longtemps avant d'avoir son itinéraire.

Un inconvénient de la replanification est le manque de réactivité advenant un temps de calcul de plusieurs minutes. Un VÉ s'ajoutant sur le réseau routier n'aura donc aucun itinéraire accessible pendant la production de la solution. De plus, les changements des itinéraires qui pourraient être demandés aux autres VÉ pourraient être communiqués avec un délai.

4.2 Contrainte d'engagement

Durant l'exécution, lorsqu'un VÉ a initié un déplacement sur une arête α entre deux bornes, la replanification doit considérer ce déplacement comme un engagement non modifiable. Autrement dit, un VÉ à mi-chemin sur α ne peut se téléporter sur une autre arête α'.

De manière générale, l'engagement est un hyperparamètre. Par exemple, au lieu de contraindre l'engagement à l'arête parcourue, on pourrait déterminer qu'un VÉ est engagé seulement à partir du moment qu'il a parcouru une portion (ou une distance) de l'arête. Cela signifierait qu'un VÉ qui de quitte une arête pourrait recevoir l'ordre de changer de borne de destination. De manière générale, on devrait pouvoir trouver de meilleures solutions de cette manière, mais le graphe des bornes n'est plus utilisable tel que défini : un VÉ peut devoir recalculer son itinéraire de n'importe quelle position sur la carte routière.

4.2.1 Impact de la contrainte d'engagement

Il est utile de comparer un engagement complet par rapport à aucun engagement. Un engagement complet signifie qu'un VÉ ne peut changer aucunement son itinéraire une fois qu'il est calculé. Ceci revient à exécuter PNCAR et à ne jamais modifier l'itinéraire. Cette inflexibilité limite les optimisations possibles de la solution globale.

À l'autre extrême, sans engagement, trouver une meilleure solution est possible, mais les itinéraires oscillent, c'est-à-dire que les plans locaux produits sont modifiés pendant l'exécution de ceux-ci. Soulignons qu'à chaque changement d'itinéraire, il n'y a aucune garantie que le conducteur va changer de destination, particulièrement si plusieurs demandes de changement surviennent rapidement. Par exemple, dans un scénario dans lequel plusieurs VÉ s'insèrent dans un petit intervalle de temps, un VÉ pourrait devoir changer plusieurs fois sa destination dans un court intervalle de temps.

Pour montrer l'impact de la contrainte d'engagement sur une arête, PCC avec et sans engagement est comparé sur la figure 4.1. Avec engagement, α s'engage à aller à la borne $\mathfrak{b}$ dès son départ. Lorsque β quitte 10 minutes plus tard, α ne peut pas changer soudainement d'arête pour aller vers la borne $\mathfrak{d}$, ce qui détériore le plan global.

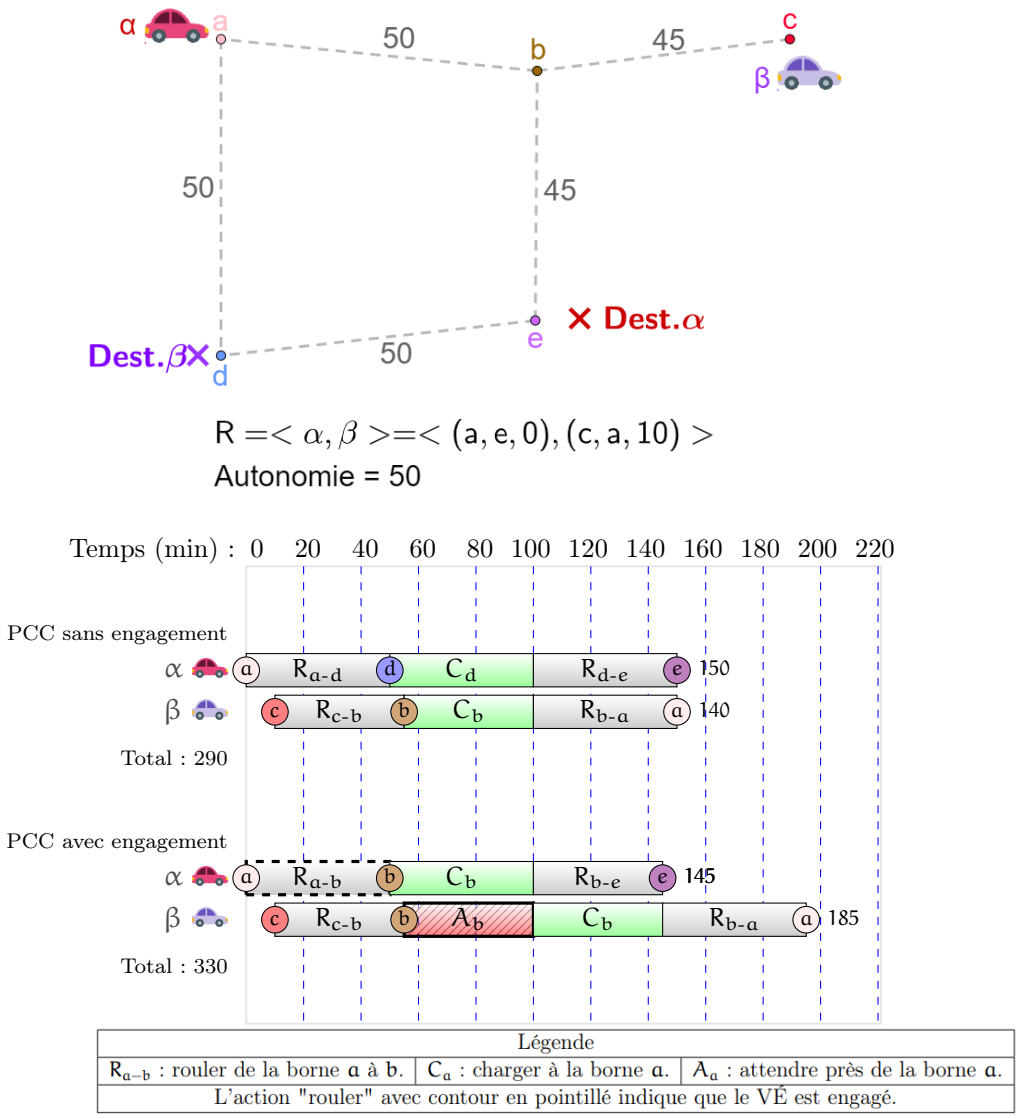


Figure 4.1 Comparaison de PCC sans et avec engagement.

4.3 Méthode PCARF : Planification coopérative avec réservation fenêtrée

Une des limitation des planificateurs coopératifs explorés dans ce mémoire est que ces méthodes n'explorent pas comment les itinéraires des VÉ peuvent s'entrelacer. Une solution que PCC génère comme dans l'exemple 3.5 ne peut être trouvée de

cette manière : le premier VÉ inséré doit faire un détour qui le ralentit pour laisser la place aux VÉ subséquents.

L'algorithme PCARF permet les entrelacements dans une certaine mesure : la méthode PNCAR est utilisée, mais lors du calcul d'un itinéraire individuel, les réservations sont effectuées jusqu'à ce que le temps d'un itinéraire dépasse un temps $\tau_{horizon} = \tau + \delta_{horizon}$. Après $\tau_{horizon}$, le calcul de l'itinéraire continu, mais avec PNC. L'ensemble des VÉ coopère, mais seulement jusqu'à la fenêtre de temps $\tau_{horizon}$. Cette procédure est décrite dans l'algorithme 11.

Algorithme 11 : Calcul de PNCAR_Horizon.

Entrée : G_b : Graphe des bornes.
$R = \langle r_1, r_2, ..., r_{|R|} \rangle$: suite de requêtes.
Sortie : π : Plan de la flotte de VÉ.

```
1  Fonction PNCAR_Horizon(G_b, R, τ_horizon) :
2      π ← ∅
3      pour r = (c_o, c_d) ∈ R faire
4          π_r ← A*Modifié(c_o, c_d, G_b)
5          π ← π ∪ {π_r}
6          //Enlève les actions qui débutent après τ_horizon de π_r.
7          π_r_horizon ←TronquerPlanDeFlotte(π_r, τ_horizon)
8          ReserverAgenda(π_r_horizon, Γ)
9      //Réserve et génère les conflits après l'horizon τ_horizon
10     ReserverAgenda(π, Γ)
11     retourner π
```

On répète ensuite le processus en avançant dans le temps : la limite de temps $\tau_{horizon}$ devient $\tau_{horizon'} = \tau_{horizon} + \delta_{avancement}$ et un nouveau vecteur de requêtes R' est créé en prenant la position des VÉ à $\tau_{horizon'}$. L'algorithme exécute ensuite PNCAR jusqu'à $\tau_{horizon'}$. Tant qu'il reste des VÉ à planifier, PNCARF continu. Cette procédure est formellement détaillée dans l'algorithme 12.

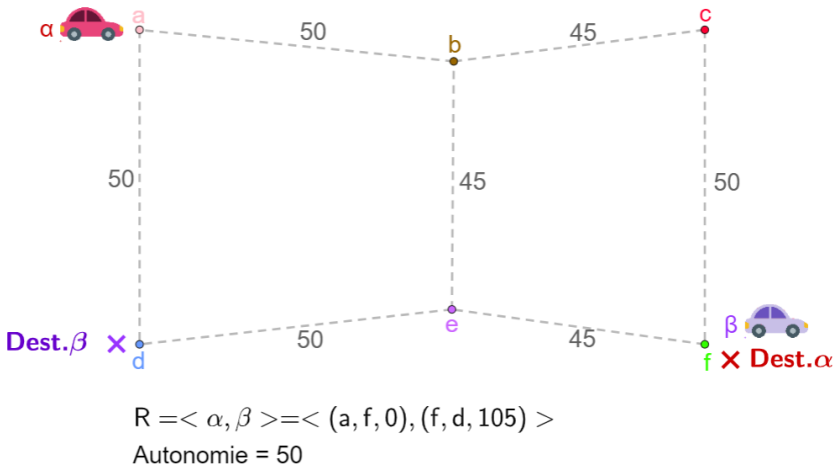
La mise en situation de la figure 4.2 compare PNCAR avec PCARF avec une fenêtre de 45 minutes. La fenêtre est considérée petite, car elle est égale ou infé-

Algorithme 12 : Algorithme PCARF.

Entrée : G_b : Graphe des bornes.
$R = \langle r_1, r_2, ..., r_{|R|} \rangle$: suite de requêtes.
Sortie : π : Plan de la flotte de VÉ.

```
1  Fonction PCARF(G_b, R) :
2      π ← ∅
3      τ ← 0
4      R' ← R
5      tant que R' ≠ ∅ faire
6          //La prochaine ligne enlève les actions qui débutent après τ de π.
7          π_partiel ←TronquerPlanDeFlotte(π, τ)
8          LibérerAgendaÀPartirDe(π, Γ, τ)
9          π ←PNCAR_Horizon(π_partiel, R, G_b, τ + δ_horizon)
10         si estInterrompu alors
11             retourner π
12         R' ← ∅
13         pour r = (b_o, b_d) ∈ R' faire
14             b' ← r.DerniereBorneQuittée(τ + δ_horizon)
15             si b' ≠ b_d alors
16                 R' ← (b', b_d)
17     τ ← τ + δ_fenêtre
```

rieure à l'arête la plus courte à parcourir. En exécutant par fenêtre le problème, PCARF réussit à résoudre l'attente à la borne c et ce, contrairement à PNCAR.



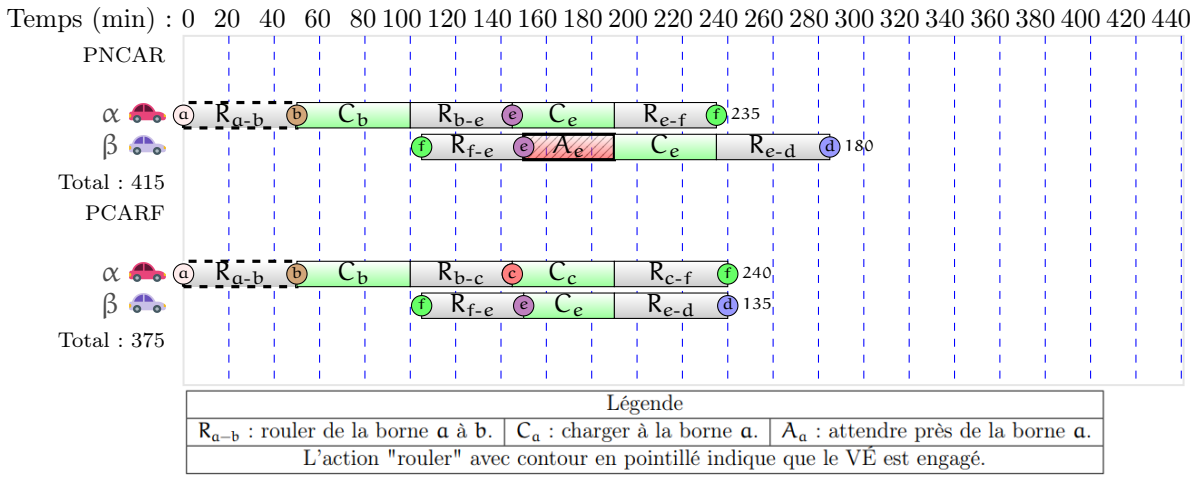


Figure 4.2 PCARF avec $\delta_{\text{horizon}} = \delta_{\text{fenêtre}} = 45$ min comparé à PNCAR.

4.4 Propriétés

Un autre avantage de PCARF est que l'algorithme 12 est possiblement interrompu à la ligne 10 : en cas d'interruption, la solution globale est généralement détériorée mais favorise l'exécution en ligne (Zilberstein, 1996). Une qualité supplémentaire

de ce planificateur est qu'il permet, à travers les paramètres δ_{horizon} et $\delta_{\text{fenêtre}}$, d'ajuster un compromis entre la qualité du plan global et le temps de calcul.

Une valeur $\delta_{\text{horizon}} \rightarrow \infty$ est l'équivalent de PNCAR. Inversement, $\delta_{\text{horizon}} \rightarrow 0$ améliore la qualité de la solution sans garantie l'obtention du plan global optimal. Le nombre de glissements de fenêtres est $\frac{\text{MpireDurée}(\pi)}{\delta_{\text{fenêtre}}}$ et représente le nombre de fois que PNCAR_Horizon doit être exécuté. La longueur de l'exécution de PNCAR_Horizon augmente avec δ_{horizon}.

4.5 Méthode PCF-P : Planification coopérative fenêtrée - permutation

PCF-P est analogue à PCF, mais l'ordre de calcul en permutation est utilisé pour chaque fenêtre de temps. On remplace donc la ligne 9 de l'algorithme 12 par PCP_Horizon qui exécute PCP en effectuant les réservations jusqu'à τ_{horizon} seulement.

4.6 Méthode PCFC : Planification coopérative fenêtrée - cascade

PCF-C est analogue à PCF, mais l'ordre de calcul en cascade est utilisé.

4.7 Méthode PCARL : Planification coopérative avec réparation locale

Les planificateurs coopératifs fenêtrés permettent de trouver de meilleures solutions que les planificateurs coopératifs non-fenêtré en plus d'être applicable dans un environnement en ligne. Cependant, déterminer les paramètres δ_{horizon} et $\delta_{\text{fenêtre}}$ est un défi. Par exemple, avec $\delta_{\text{fenêtre}} = 1$ min, les solutions seront de meilleure qualité sauf que le temps de calcul augmentera appréciablement. De plus, les variantes *permutation* et *cascade* affectent non linéairement la durée de calcul en fonction du nombre de VÉ à planifier.

Pour contourner ces limitations, une méthode inspirée du concept de la réparation de plans de (Zelinsky, 1992) et de la fusion de plans de (Beaudry *et al.*, 2005) est envisageable. Dans le contexte des VÉ, une idée originale est de d'abord trouver les conflits causant les plus grandes pertes de temps. Une perte de temps est une combinaison d'un détour et d'une attente. On tente une réparation en détournant un des VÉ impliqués dans ces conflits causant des pertes de temps. Pour formaliser le tout, les notions de *pénalité d'arête* et de *perte de temps* sont d'abord définies.

4.7.1 Activité d'un VÉ sur une arête

L'activité d'un VÉ sur une arête α est définie comme :

$$d_{\text{activité}}(\alpha) = d_{\text{deplacement}}(\alpha) + d_{\text{chargement}}(\alpha) \tag{4.1}$$

L'activité totale de l'itinéraire π_i du VÉ est la somme des activités sur les segments de l'itinéraire :

$$d_{activité}(\pi_i) = \sum_{\alpha \in K} d_{deplacement}(\alpha) + d_{chargement}(\alpha) \tag{4.2}$$

avec $K = \langle \alpha_1, \alpha_2, ..., \alpha_{|K|} \rangle$ l'ensemble des arêtes parcourues sur π_i.

4.7.2 Détour moyen d'un itinéraire de VÉ

En se rappelant la définition de l'itinéraire optimal d'un VÉ 1.9, le détour moyen d'un itinéraire π_i est :

$$\overline{d_{detour}(\pi_i)} = \frac{d_{activité}(\pi_i) - d_{activité}(\pi_i^*)}{|K^*|} \tag{4.3}$$

avec K^* le nombre d'arêtes parcourues dans π_i^*.

Lorsqu'aucun détour n'est effectué et qu'un itinéraire parcourt les mêmes bornes que l'itinéraire optimal, alors $\overline{d_{detour}(\pi_i)} = 0$ par définition. Plus le plan π_i est affecté par les détours, plus $\overline{d_{detour}(\pi_i)}$ augmente.

4.7.3 Perte de temps

Une perte de temps sur l'arête α d'un itinéraire π_i d'un VÉ est :

$$d_{perte}(\alpha) = \overline{d_{detour}(\pi_i)} + d_{attente}(\alpha) \tag{4.4}$$

avec $d_{attente}(\alpha)$ le temps d'attente.

$d_{perte}(\alpha)$ permet d'estimer la perte de temps lors du parcours d'une arête. Comme spécifié au début de ce chapitre dans la section 4.7, celle-ci dépend d'un détour et d'une attente.

4.7.4 Liste ordonnée des pertes de temps de π_i

Dans un contexte de planification en ligne, soit un nouveau VÉ s'insérant sur la route. Avec PNCAR, un itinéraire π_i est produit. Le détour moyen est calculé et la *perte de temps* est calculée pour chaque arête $\alpha_m \in K = \langle \alpha_1, \alpha_2 \cdots \alpha_{|K|} \rangle$. Les arêtes avec $d_{attente} = 0$ sont exclues de la liste. Cette liste est finalement ordonnée en ordre décroissant selon la *perte de temps* associée à chaque arête.

4.7.5 Algorithme PCARL

Dans un contexte de planification en ligne, soit un plan global π dans lequel un nouveau VÉ s'insère. PNCAR est exécuté et un itinéraire π_i est calculé. La *liste ordonnée des pertes de temps* est générée à partir de π_i. Pour chacune des *pertes de temps*, 3 plans voisins sont comparés (Ghallab *et al.*, 2016) :

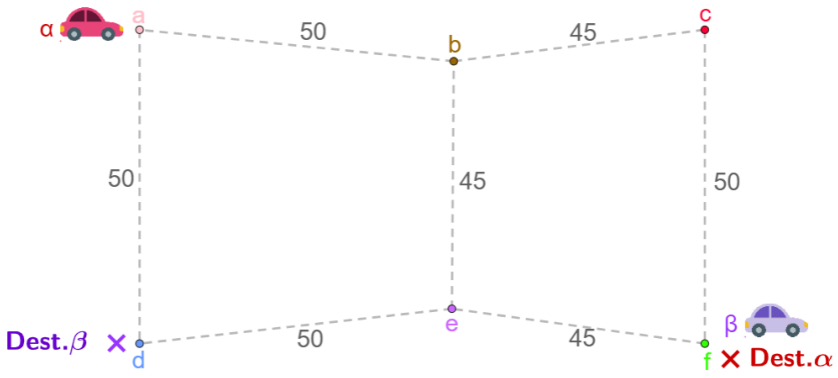
1. Laisser le conflit tel quel ;
2. ν_i est forcé d'éviter la borne b_c puis exécute PNCAR ;
3. Si ν_j n'est pas engagé à cette borne, il est forcé de détourner la borne b_c puis exécute PNCAR ;

Si l'option 2 ou 3 est choisie, le plan est conservé, la *liste ordonnée des pertes de temps* régénérée et l'algorithme reprend du début. Lorsqu'aucune amélioration n'est possible, l'algorithme termine : un optimal local a été trouvé.

Une explication formelle est fournie par l'algorithme 13. Un avantage de PCARL est qu'il est possible de l'interrompre à la ligne 14 en ayant une solution valide dès la première itération, favorisant ainsi la planification en ligne (Zilberstein, 1996).

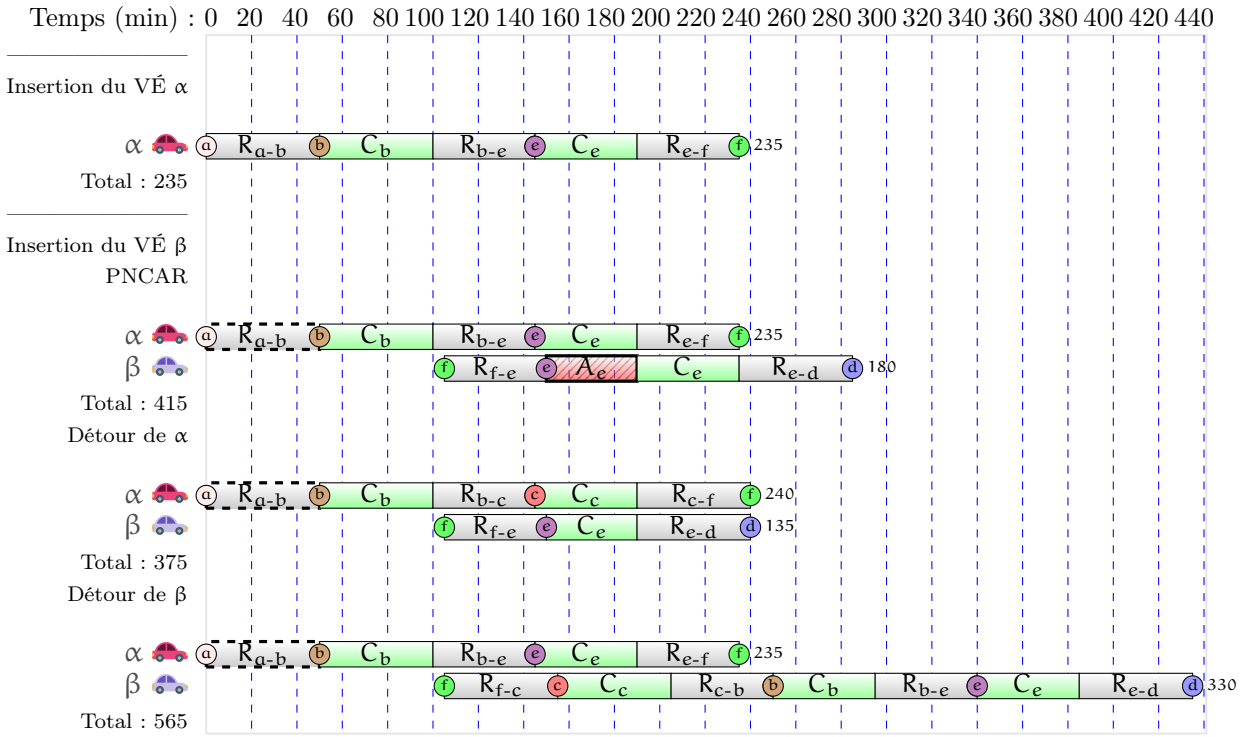
4.8 Mise en situation

Une mise en situation est illustrée sur la figure 4.3. Le VÉ α s'insère sur la route et s'engage sur l'arête (a, b). Le VÉ β quitte à 105 minutes de la borne f. En utilisant PNCAR, il y aura un unique conflit à la borne e. On tente de le réparer en forçant un détour de α de la borne e. La solution globale est améliorée par rapport à PNCAR. Un détour de β de la borne e est également évalué, mais la solution est détériorée par rapport à PNCAR. La solution qui consiste à forcer un détour par α est donc conservée.



$R =< \alpha, \beta >=< (a, f, 0), (f, d, 105) >$

Autonomie = 50



Légende		
R_{a-b} : rouler de la borne a à b.	C_a : charger à la borne a.	A_a : attendre près de la borne a.
L'action "rouler" avec contour en pointillé indique que le VÉ est engagé.		

Figure 4.3 Exemple de mise en application de PCARL avec deux VÉ.

Algorithme 13 : Calcul de PCARL .

Entrée : G_b : Graphe des bornes.
$R = \langle r_1, r_2, ..., r_{|R|} \rangle$: suite de requêtes.
Sortie : π : Plan de la flotte de VÉ.

```
1  Fonction PCARL(G_b, R) :
2      π ←PNCAR(R, G_b) //On peut utiliser PNCAR ou PNC.
3      EstAmélioré ← Vrai
4      tant que EstAmélioré faire
5          EstAmélioré ← Faux
6          Z ← TrouveConflits(π, G_b)
7          pour Conflit z ∈ Z faire
8              π_nouveau ← MeilleureAlternative(π, z, G_b)
9              si M(π_nouveau) ≤ M(π) alors
10                 π ← π_nouveau
11                 EstAmélioré ← Vrai
12                 si estInterrompu alors
13                     retourner π
```

4.9 Variantes de PCARL

Il y a trois dimensions différentes explorées dans ce mémoire, chacune ayant deux possibilités, créant $2^3 = 8$ variantes.

4.9.1 PNCAR ou PNC

La ligne 3 de l'algorithme 13 peut être remplacée par PNC . Le détour moyen est alors 0 min lors de l'insertion d'un VÉ car l'itinéraire est celui calculé par A*. Cette variation implique que le plan individuel du VÉ inséré n'est pas préalablement optimisé par PNCAR.

Glouton ou sobre

La version gloutonne de l'algorithme est celle déjà présentée : dès qu'une amélioration du plan global est possible à la ligne 10 de l'algorithme 13, le calcul reprend partir de la nouvelle solution globale choisie. La version sobre énumère tous les voisins créés à partir de la liste ordonnées des pertes de temps et seulement ensuite sélectionne le meilleur plan. Cette version de PCARL est détaillée dans 14. La version sobre est en général plus longue à exécuter car elle nécessite la création et l'évaluation d'un plus grand nombre de voisins.

Algorithme 14 : Calcul de PCARL sobre

Entrée : G_b : Graphe des bornes.
$R = \langle r_1, r_2, ..., r_{|R|} \rangle$: suite de requêtes.
M : Métrique (fonction objectif).
Sortie : π : Plan de la flotte de VÉ.

```
1  début
2      π ←PNCAR(R, G_b, M) //Peut être substituée par PNC.
3      EstAmélioré ← Vrai
4      tant que EstAmélioré faire
5          pour Conflit z ∈ Z faire
6              π_nouveau ← si M(π_nouveau) ≤ M(π_meilleur) alors
7                  EstAmélioré ← Vrai
8                  π_meilleur ← π_nouveau
9          π ← π_meilleur
```

4.9.2 Incrémentale ou entière

La version incrémentale est celle déjà présentée : les *pertes de temps* sont calculées uniquement à partir du nouveau VÉ inséré. Dans la version entière, on exécute PNC pour tous les VÉ de la requête et non seulement le dernier. Ensuite, la liste des pertes pour tous les VÉ est établie et on tente de résoudre les conflits. Ceci permet de partir d'une solution globale non-optimisée préalablement par PCARL.

La version entière est plus longue à exécuter que la version incrémentale car elle calcul les voisin à partir des itinéraires de tous les VÉ.

4.9.3 Nomenclature sur PCARL

Il existe 3 variantes de PCARL , avec 2 alternatives chacune, ce qui créé $2^3 = 8$ variantes au total. Un système de nomenclature abrégé est proposé dans la table 4.1 permettant ainsi de s'y retrouver.

PNCAR ou PNC	Glouton ou sobre	Incrémental ou entier
PNCAR	G	I
PNC	S	E

Tableau 4.1 Infixe désignant chaque variation de PCARL. PCARL-PNCAR-G-I désigne PCARL glouton incrémental utilisant PNCAR.

CHAPITRE V

EXPÉRIMENTATION

Le chapitre sur les expérimentations souligne les détails de la suite de logiciels crées pour tester les planificaturs, des données utilisées, de la génération de scénarios ainsi que de l'environnement.

5.1 Architecture de l'environnement d'évaluation

La suite de logiciels *CoRoadperation* a été développée expressément pour lire des cartes routières (incluant les bornes de recharge), la génération de scénario, la planification et les tests de comparaison. Le code source est ouvert et publiquement accessible [1]. Ce projet a été développé dans l'optique d'expérimenter sur les algorithmes, mais aussi d'améliorer les performances d'une application interactive de planification d'itinéraire de véhicules électriques nommée EvNav [2].

La figure 5.1 présente les modules de *CoRoadperation*. Le langage de programmation de *CoRoadPeration* est C++ à l'exception du *visualisateur de plans* qui est implémenté en Java.

---

1. `https://bitbucket.org/akilea/coroadperation`

2. https ://github.com/giraldeau/evnav

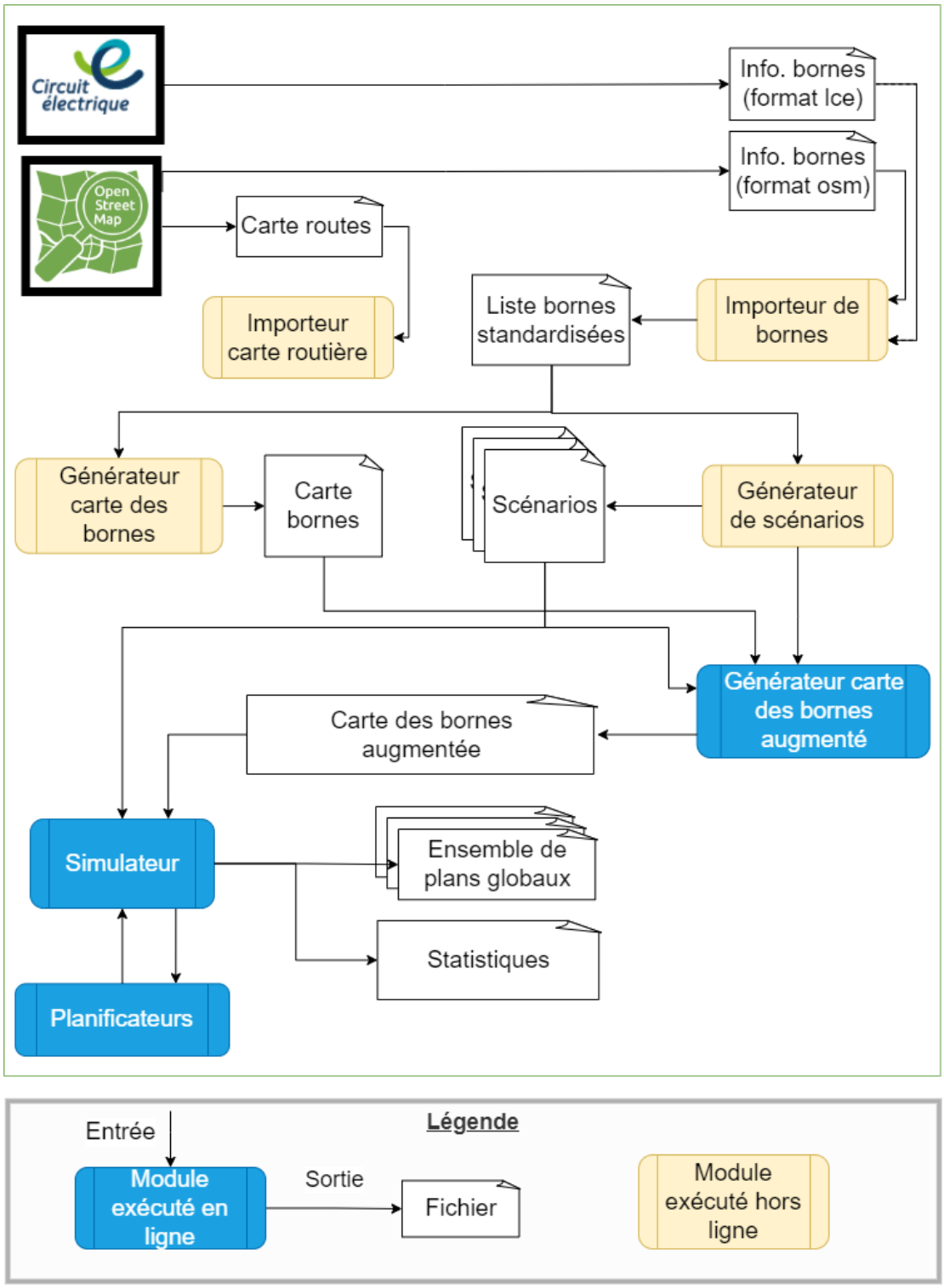


Figure 5.1 Architecture de CoRoadperation.

5.2 Données utilisées

La sélection des données de test est effectuée à l'instar de référence dans le domaine de la recherche sur l'optimisation du calcul d'itinéraire de VÉ. Certains chercheurs évaluent leurs planificateurs sur des données issues de OpenStreetMap(OpenStreetMap Foundation, 2022), une carte ouverte dont les données sont disponibles sous la licence libre de carte interactive [3] (figure 5.2), pour la région de Bavaria, un état d'Allemagne (Sachenbacher *et al.*, 2011). De manière analogue, d'autres chercheurs font des simulations pour des itinéraires sur la carte de la France (De Nunzio *et al.*, 2020). Pour ce mémoire, la carte routière de la province de Québec (section 1.2.1) $\mathsf{G} = (\mathsf{S}, \mathsf{A})$ a été extraite depuis le projet OpenStreetMap.

Le service de consultation de la carte est utilisé. *647 391* routes (arêtes) et *3 529 739* intersections de la province de Québec sont disponibles en format OpenStreetMap (*osm*) en date du 19 avril 2021 (OpenStreetMap Foundation, 2022). Le format *OSM* fourni une liste de coordonnées d'intersections ainsi qu'une liste des paires d'intersection liées, formant ainsi une carte routière complète. Les coordonnées des bornes de recharge sont également présentes.

5.2.1 Données d'un réseau de bornes de recharge

Dans le but de comparer les différents planificateurs, trois jeux de données de tailles croissantes sont utilisés. Le *petit réseau* désigne 45 bornes de recharge fournies par OSM : il représente la distribution de bornes en 2014 et permet d'étudier le passé. Le *moyen réseau* est composé des 637 bornes de recharge rapide extraites du site web de Circuit Électrique (`lecircuitelectrique.com`), une di-

3. OpenStreetMap est sous la licence Open Data Commons Open Database License (ODbL).

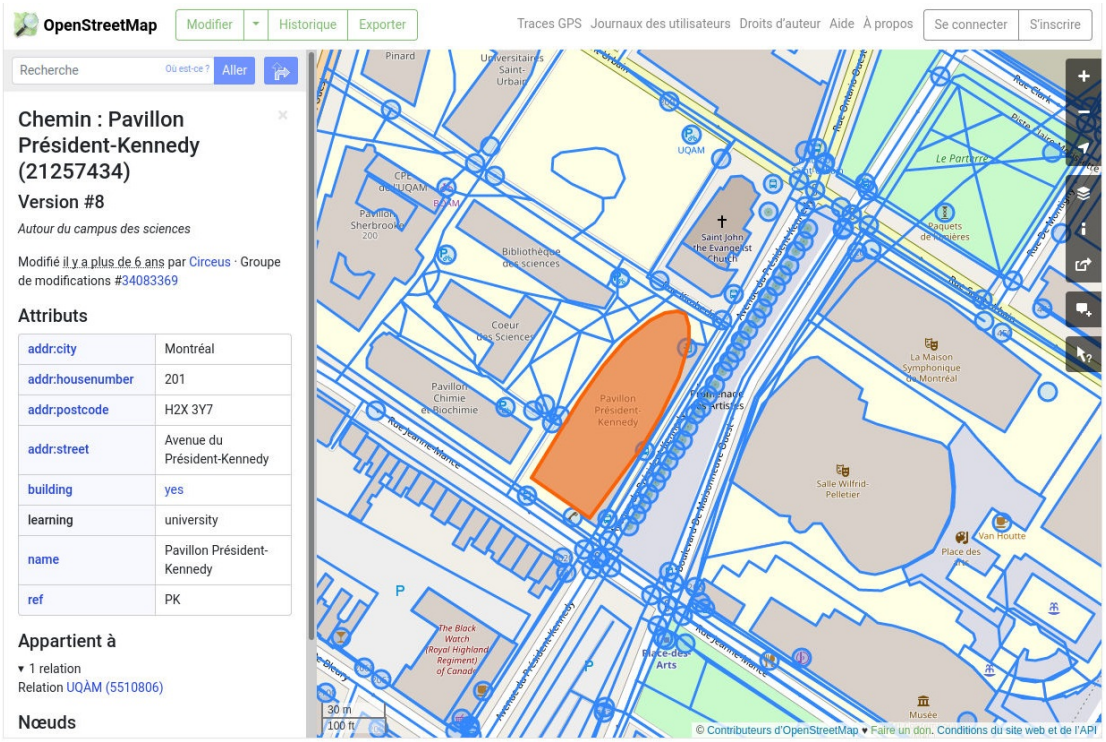


Figure 5.2 Exemple d'une carte affichée dans l'interface utilisateur du projet OpenStreetMap (OpenStreetMap Foundation, 2022)

vision d'Hydro-Québec responsable de compiler et rassembler des informations liées au réseau de bornes de recharge du Québec. Finalement, le *gros réseau* représente 3364 bornes combinant toutes les bornes de recharge rapides et lentes du site de Circuit Électrique. Cette configuration représente une estimation du développement futur du réseau de bornes.

Toutes les données des bornes ont été exportées le 19 avril 2021. Les sources d'information à propos des bornes étant variées, le module d'importeur de bornes extrait les informations des différentes sources et les transforme dans un format CSV uniforme (exemple fourni dans l'annexe 6.2).

5.3 Générateur du graphe des bornes

Ce générateur du graphe des bornes implémente la section 1.2.3. Il prend en entrée la carte routière du Québec d'OSM, un fichier uniforme de la liste des bornes et produit en sortie un fichier de graphe des bornes. Il est exécuté hors ligne comme indiqué sur la figure 5.1 de l'architecture. Un échantillon du format du graphe des bornes est donné dans l'annexe 6.3.

5.4 Générateur de scénarios

Pour générer des requêtes réalistes, le générateur utilise une liste de modèles de VÉ. Cette liste reprend les VÉ les plus vendus au Québec.

5.4.1 Autonomie des VÉ

Tel de mentionné dans la section 1.1.1, l'autonomie est fonction du modèle de VÉ. Deux profils de flottes de VÉ sont étudiés : une flotte composée uniquement du modèle Hyundai Ioniq en hiver (146km d'autonomie) et l'autre flotte hétérogène est composée des modèles décrits dans la table 5.1 et ce, en hivers (Société d'Assurance Automobile du Québec et Association Véhicules Électrique du Québec, 2021). Noter que la fréquence de leur présence sur la route est utilisée dans le simulateur afin d'avoir des données près de la réalité. L'autonomie en hivers est utilisée pour avoir de petites autonomies et maximiser l'utilité des planificateurs.

Modèle	Part de marché	Autonomie (km)	Autonomie hivernale (km)
Tesla Model 3	25	352	259
Chevrolet Bolt	21	381	280
Nissan Leaf	19	171	126
Hyundai Kona Électrique	13	302	223
Hyundai Ioniq	6	198	146
Volkswagen Golf-e	6	200	147
Kia Soul EV	6	149	110
Tesla Model Y	4	384	283

Tableau 5.1 Caractéristiques des modèles de VÉ les plus populaires au Québec.

5.4.2 Générateur de requêtes

Le générateur de requêtes prend en paramètres :

1. une liste de zones de départ ;
2. une liste de zones d'arrivées ;
3. un intervalle de temps ;
4. une liste pondérée des autonomies de VÉ (table 5.1) ;
5. un nombre de VÉ à créer.

Une zone de départ et une zone d'arrivée sont choisies. Un générateur échantillonne des distributions uniformes pour trouver les positions d'origine et de destination à l'intérieur de ces zones. Le système vérifie qu'un itinéraire existe : autrement, de nouveaux points sont tirés.

Un heure de départ est échantillonnée dans une distribution uniforme bornée par intervalle de temps donné en paramètre. Une autonomie est échantillonnée selon la part de marché en utilisant la liste de la section 5.1. À la fin, la liste de requêtes est ordonnée de manière croissante selon le temps de départ. Intuitivement, cette procédure permet, par exemple, de créer 500 VÉ qui partent de Montréal entre 12h00 et 14h00 pour se rendre à Matane. Deux scénarios sont étudiés dans ce mémoire :

le scénario A, Montréal-Matane et le scénario B, Montréal-Matane+Saguenay-Sherbrooke. La figure 5.3 aide à visualiser les zones. L'annexe 6.4 en donne les coordonnées exactes des limites de zones.

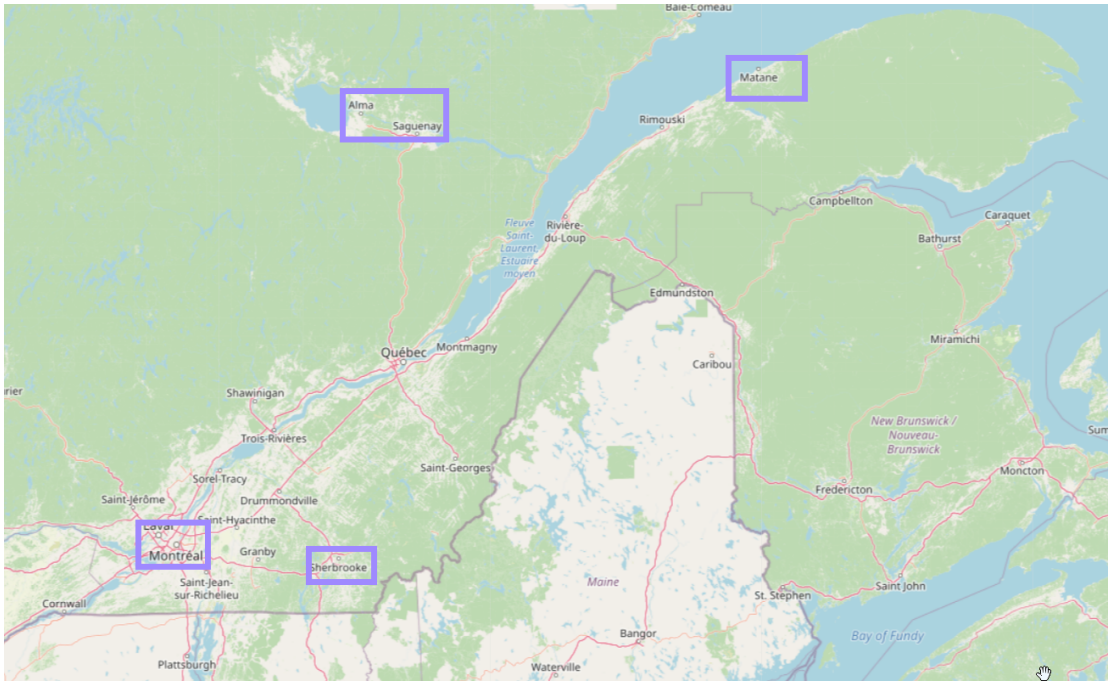


Figure 5.3 Illustration des zones de départs et d'arrivées utilisées pour créer les scénarios.

5.4.3 Scénarios

Les scénarios ont été désignés pour étudier les itinéraires nécessitant au moins 2 recharges entre des villes importantes du Québec pour le VÉ ayant la plus grande autonomie (*Tesla Model Y* avec 283 km en hiver). Le scénario A utilise les zones Montréal-Matane pour créer les départs et arrivées, créant ainsi un flux de VÉ dans chaque direction. Le scénario simule le déplacement d'une flotte de VÉ entre deux grands centres du Québec. Ce scénario simplifie l'analyse des résultats, mais n'est pas réaliste : des départs s'effectuent à différents endroits, typiquement davantage

dans les villes. Le scénario B, avec 4 flux de VÉ circulant en forme de croix entre 4 villes, simule une flotte de VÉs distribuée davantage à travers le province.

Les deux scénarios utilisent une valeur aléatoire du temps de départ pour chaque VÉ entre 0h et 2h correspondant à la durée de l'heure de pointe entre Montréal et la ville de Québec (Données Québec, 2022). Un échantillon de requêtes générées est fourni dans l'annexe 6.5. Pour chaque scénario, dix listes de requêtes sont générées dans le but de tester la stabilité des planificateurs.

5.5 Planificateurs évalués

Les planificateurs évalués sont décrits dans la table 5.2 et une description des paramètres est incluse.

Abréviation	Planification...	Paramètres/Variations
PI	...individuelle	-
PNC	...non coopérative	-
PNCAR	...non coopérative avec réservation	-
PCC	...coopérative complète	-
PC	...coopérative	Permutation (P) ou cascade (C)
PCF	...coopérative fenêtrée	δ_{horizon} : durée d'horizon. (3h,2h et 1h) $\delta_{\text{déplacement}}$: déplacement de la fenêtre (1.5h, 1h et 0.5h) Permutation (P), cascade (C) ou ordre d'arrivé (O)
PCARL	...coopérative avec réparation locale	PNCAR ou PNC Glouton ou sobre Incrémental ou entier

Tableau 5.2 Planificateurs et paramètres évalués.

Tous les planificateurs peuvent compléter un plan global partiel.

5.6 Simulateur

Le simulateur est le module applicatif du projet : il transmet la liste de requêtes aux planificateurs puis enregistre les résultats. Quelques utilitaires sont fournis,

comme la copie de plans et la copie de circuit électrique accompagné de sa table de réservation. Le simulateur peut être utilisé en mode de départs simultanés (planificateurs hors ligne) ou en mode de départs continus (planificateurs en ligne).

Outre les plans, plusieurs artefacts sont en sortie du simulateur.

— Description itinéraire : Décrit en détail un plan local (exemple dans l'annexe 6.6) ;
— Sommaire des plans globaux : Liste des résultats des plans globaux (exemple dans l'annexe 6.14) ;
— Exporteur résultat : Écrit en détail les paramètres et résultats des plans globaux dans un fichier CSV (exemple dans l'annexe 6.7).

5.7 Environnement d'exécution

CoRoadperation a été exécuté sur un ordinateur équipé d'un processeur *Intel Core i7-10700* avec 64 Go de mémoire vive. Le temps de calcul pour chaque algorithme a été limité à 5 minutes (300 secondes). Tout algorithme qui dépasse ce temps de calcul ne rapporte un résultat sous-optimal et est désactivé. Dans ce mémoire, les solutions issues des algorithmes désactivés ne sont pas comparés et sont ignorés dans l'analyse finale.

5.8 Résumé des paramètres expérimentaux

Pour faciliter la compréhension, un résumé des paramètres expérimentaux est sur la table 5.3.

Le scénario A (Montréal-Matane) avec une flotte de VÉ uniforme composée uniquement du modèle Ioniq 5 en hiver sur un réseau moyen sera principalement étudié lors des discussions du prochain chapitre (section 6). Minimiser l'autono-

Paramètre	Valeur(s)
Scénario	A : Montréal-Matane B : Montréal-Matane+Saguenay-Sherbrooke
Réseau de bornes	Petit (45 bornes) Moyen (637 bornes) Grand (3364 bornes)
Autonomie (composition de la flotte de VÉ)	Ioniq en hivers seulement (146 km) Distribution du tableau 5.4
Nombre de scénarios créés par ensemble de paramètres	10
Intervalle de départ	2 h
Fonction objective	Pénalité
Temps planification maximal (exclu calcul graphe des bornes augm.)	5 minutes

Tableau 5.3 Paramètres expérimentaux.

mie des VÉ en étudiant seulement la Ioniq 5 fera valoir davantage la différence entre les planificateurs et simplifie la compréhension. Le réseau de bornes moyen est choisi, car le petit réseau contient trop peu de bornes et le réseau de bornes devient rapidement congestionné. Finalement, dix échantillons sont générés et testés par scénario.

CHAPITRE VI

RÉSULTATS ET DISCUSSION

Les planificateurs hors ligne sont discutés suivit des planificateurs en ligne. Un examen des différentes fonctions objectives et de l'autonomie conclu le chapitre.

6.1 Planificateurs hors lignes

Les pénalités des planificateurs hors lignes se trouvent sur la figure 6.1 et les durées de calcul, sur la figure 6.2.

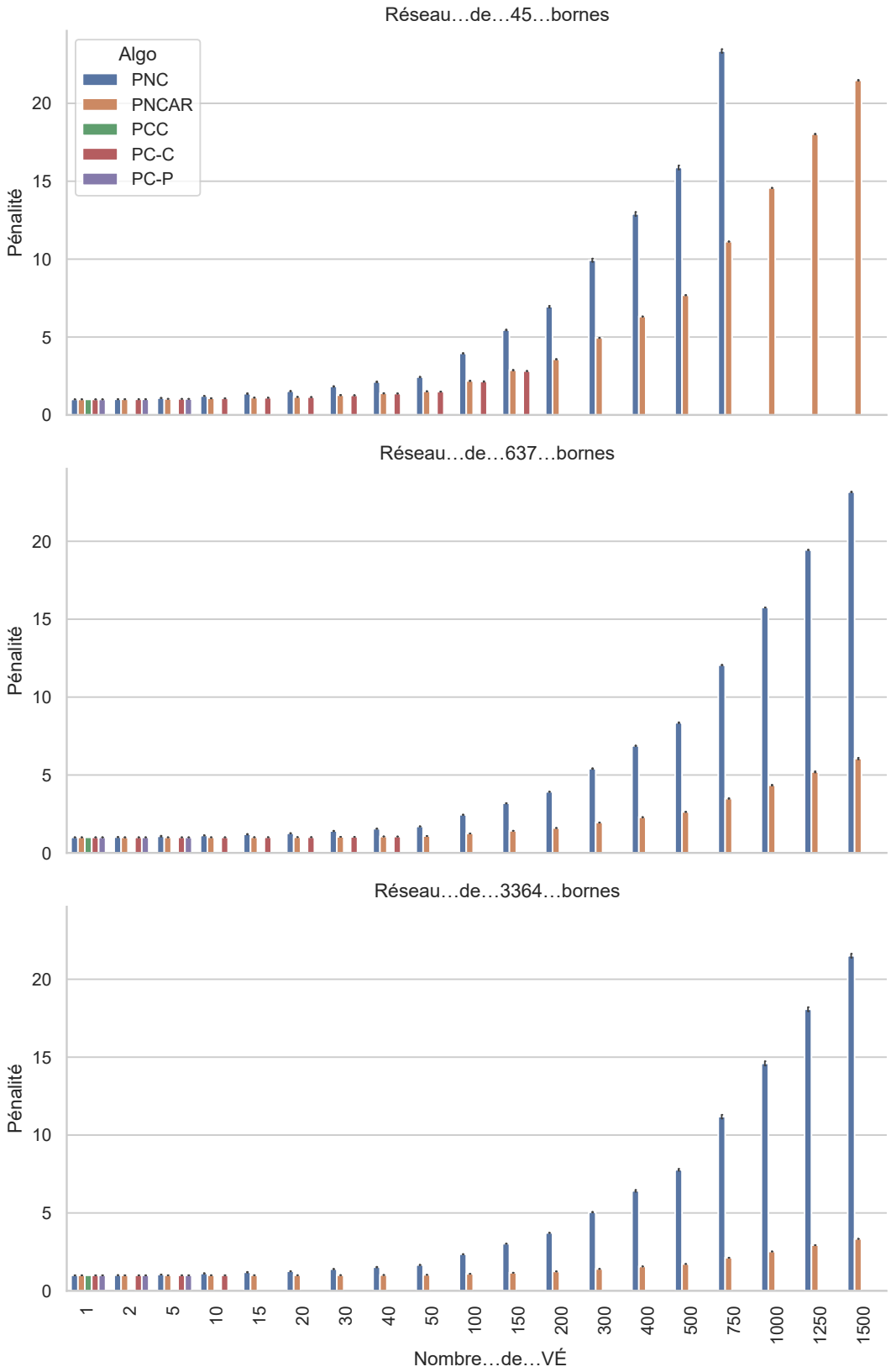


Figure 6.1 Pénalités des planificateurs hors lignes

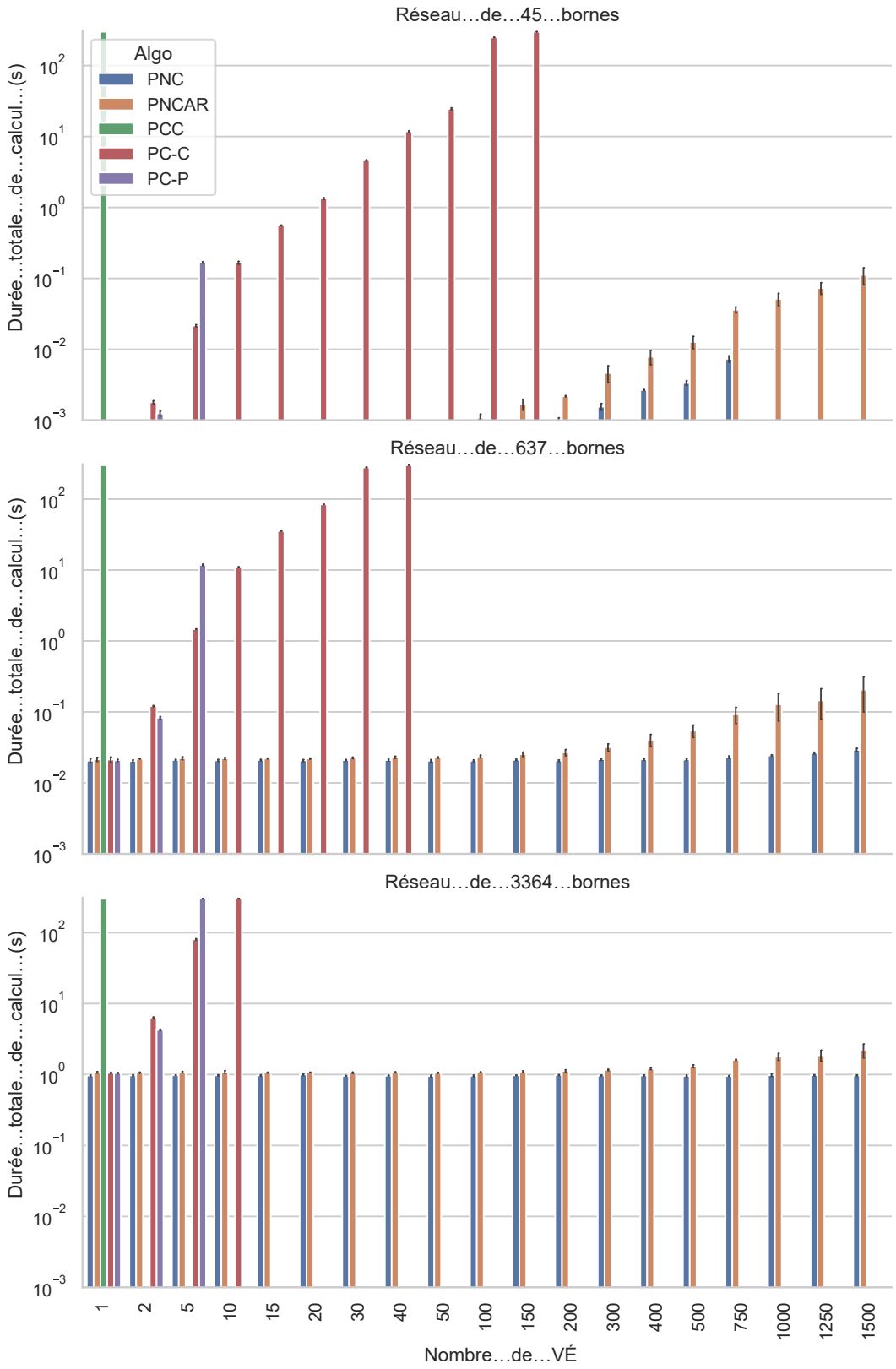


Figure 6.2 Durée de calcul moyen sur 10 essais des planificateurs hors lignes

Rappelons que 10 essais sont générés : les résultats présentés sont la moyenne de ces 10 essais. La présence d'écart-type est illustrée à l'aide d'une ligne verticale. Une première remarque est que l'écart-type est difficilement visible, car généralement, il représente moins de 2% dans les pires cas, comme pour la pénalité et la durée de calcul de PNC à partir de 1000 VÉ sur le réseau moyen.

6.1.1 Qualité des plans générés

PNC affiche la pire pénalité pour toute quantité de VÉ. C'est le résultat attendu : les VÉ cherchent à optimiser leurs itinéraires individuellement sans coopérer.

PNCAR est une première planification optimisée sans coopération. Les solutions générées par PNCAR ont systématiquement une pénalité plus basse ou égale à PNC, ce qui est également un résultat attendu. Cette solution est utile pour établir une limite supérieure : un planificateur coopératif arborant des pénalités supérieures est inintéressant. Pour le réseau moyen et 500 VÉ, la pénalité est de 2.6 pour PNCAR comparé à 8.1 pour PNC. Dans les deux cas, il est pertinent de se demander si le réseau de bornes est surchargé et si le nombre de VÉ est excessif.

PCC est introuvable sur le graphique. La raison est que pour un seul VÉ, le temps de calcul de 5 min est dépassé, et ce, même pour le petit réseau. Un scénario a été évalué avec 72h de temps de calcul mais le calcul n'était pas terminé. Cette méthode devrait générer de meilleures solutions, mais il est difficile de le constater dans ce mémoire.

PC-C, le planificateur coopératif le plus simple, n'affiche aucun résultat au-delà de 40 VÉ : le temps de calcul maximum de planification alloué a été dépassé avant d'atteindre 50 VÉ et ce, pour les 10 essais. Aucune amélioration n'est notée avant l'insertion de 20 VÉ, mais une amélioration de 3,4 % est notée pour le 40e VÉ

calculé.

6.2 Durée de calcul

Les durées de calcul de PNC et PNCAR sont inférieures à une seconde, et ce, peu importe le nombre de VÉ : ces algorithmes étant non coopératifs, la complexité augmente peu selon le nombre de VÉ. À titre indicatif, sur le réseau moyen avec 1500 VÉ, PNC prend 0,12 s à calculer et PNCAR, 0,52 s. PNCAR est plus long à calculer, car des inscriptions et consultations sont effectuées dans l'agenda. On doit ajouter également la durée de calcul du chronométrage qui se fait à chaque noeud exploré par *A*Modifié*.

PCC n'est pas présent, car la durée dépasse cinq minutes même pour un seul VÉ. Des tests sur des mises en situation plus simples ou encore augmenter le temps de calcul est nécessaires pour tirer une conclusion. La durée de calcul de PC-C dépasse la durée maximum de calcul à 150 VÉ sur le petit réseau, 50 VÉ pour le réseau moyen et 15 VÉ sur le grand réseau. PC-C est l'équivalent d'exécuter PNCAR à $|\mathsf{R}|^2$ reprise et la durée de calcul augmente considérablement à mesure que le nombre de VÉ augmente. Il serait intéressant d'augmenter le temps de simulation ou de tester sur un problème plus simple afin de comprendre comment l'amélioration évolue.

Au-delà de 5 VÉ, la durée maximale de calcul est dépassée pour PC-P. Le temps de calcul augmente en $|\mathsf{R}|!$, ce qui rend PC-P impraticable. Les quelques données récoltées ne suggèrent aucune amélioration du temps de déplacement : elles sont identiques à PNCAR et PC-C, mais comme avec PC-C, tester sur un problème plus simple ou augmenter la durée de calcul allouée pourrait révéler une amélioration.

6.2.1 Conclusion

Les planificateurs coopératifs inspiré de travaux précédents ne présentent pas autant de gain de performance (Silver, 2005). Il y a deux raisons à cela : la première est que les auteurs effectuent des tests en planifiant des itinéraires sur une grille sur laquelle les agents peuvent seulement bouger dans les quatre directions cardinales (se référer à la figure 2.2 pour un rappel). Ceci implique que quatre noeuds (ou moins sur les bords de la carte) sont accessibles par un agent. Pour ce mémoire, en se référant à la carte moyenne comme exemple, un modèle Ioniq 5 quittant l'UQÀM à Montréal a accès à 112 bornes de recharge sur les 647 sont utilisées. En contrepartie, une petite partie (28 pour PNCAR avec 1500 VÉ sur un réseau de taille moyenne) des bornes est empruntée. Autrement dit, le degré de chaque noeud est élevé comparé à (Silver, 2005), ce qui explique le temps de calcul important.

Une deuxième raison est liée au nombre de noeuds formant un itinéraire. Les auteurs des précédents travaux génèrent itinéraire de 22 noeuds au minimum pour un agent seul sur la carte. Dans ce mémoire, le nombre de noeuds est 2 au minimum pour un seul VÉ. Ces itinéraires relativement petits limitent l'opportunité de coopérer entre les VÉ. PNCAR présente des résultats généralement intéressants sans aucune coopération.

Pour mieux faire ressortir les forces des algorithmes étudiés dans ce mémoire, il est nécessaire d'explorer un problème avec moins de bornes sur de plus longues distances pour se rapprocher des conditions de (Silver, 2005) et mieux constater la différence entre les algorithmes. Augmenter la durée maximum de calcul permise permettrait aussi de mieux constater la performance des algorithmes plus gourmands. Cependant, ces algorithmes ont une applicabilité limitée dans une application réelle étant donnée leur temps de calcul considérable.

Étude de cas de PC-C

Afin d'avoir une meilleure visibilité sur l'exécution de PC-C, une trace de calcul (figure 6.3) a été générée. Le graphique se lit de gauche à droite : chaque point bleu (l'étiquette "Itération") représente la qualité du plan pour l'un des 220 ordres de calcul pour PC-C. La meilleure solution trouvée jusqu'à présent est indiquée avec l'étiquette "Meilleure". PNCAR est mis en référence.

Le nombre d'itérations est de $\frac{21\times20}{2} = 220$ itérations (explications dans la section 2.3.3). La présence des solutions équivalentes à des travaux existants s'observe : plusieurs itérations donnent exactement la même pénalité que (Latombe, 1991). Une grande variété de valeurs de solutions sont trouvées, mais la différence entre la pire solution (36298 s) et la meilleure (36229 s) est de 69 s, soit 0,19% d'écart.

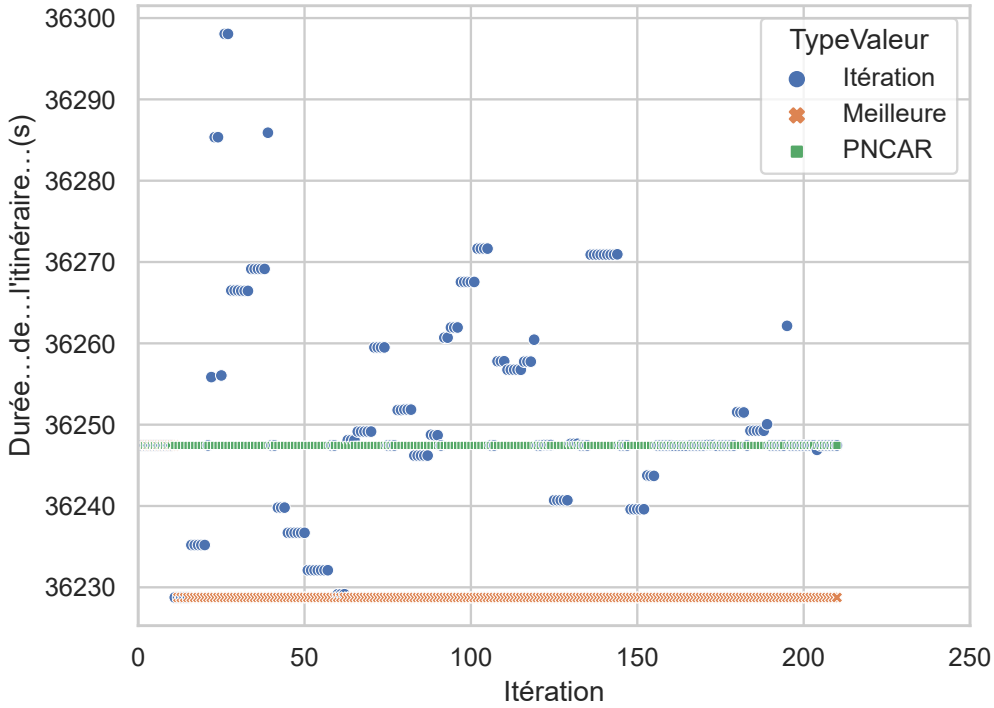


Figure 6.3 Traces du calcul de PCNC avec 20 VÉ de l'expérience 1

6.3 Planificateurs en ligne

Vu le nombre de planificateurs testés, cette section présente d'abord les planificateurs coopératifs fenêtrés suivit des planificateurs basés sur la réparation locale.

6.3.1 Planificateurs coopératifs fenêtrés

Sur les figures 6.4 et 6.5 sont présentées les pénalités pour trois grandeurs de fenêtres de temps. PNCAR a été inclu comme référence. La durée de calcul associée se trouve sur les figures 6.6 et 6.7.

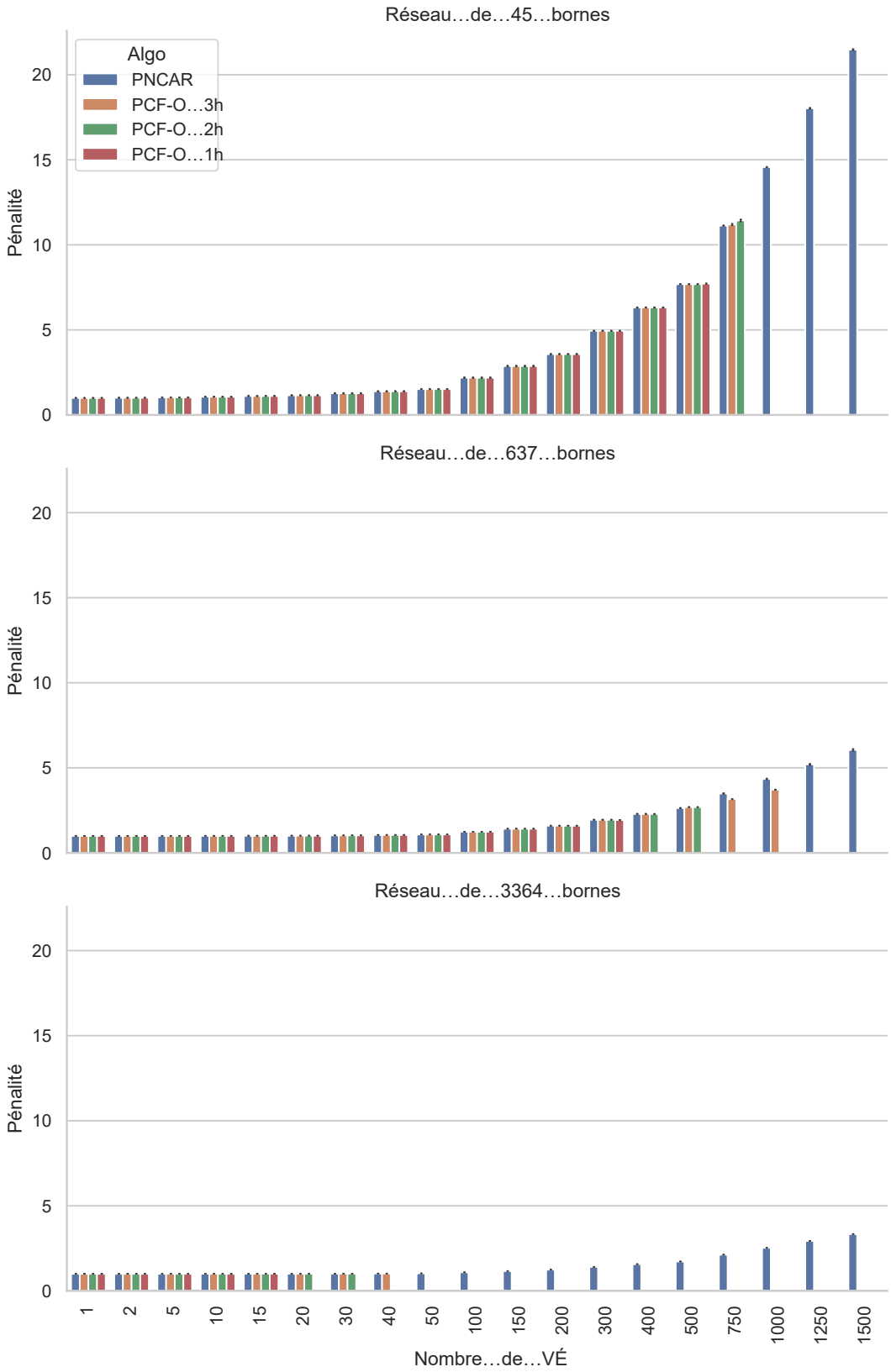


Figure 6.4 Pénalités des planificateurs en ligne fenêtrés

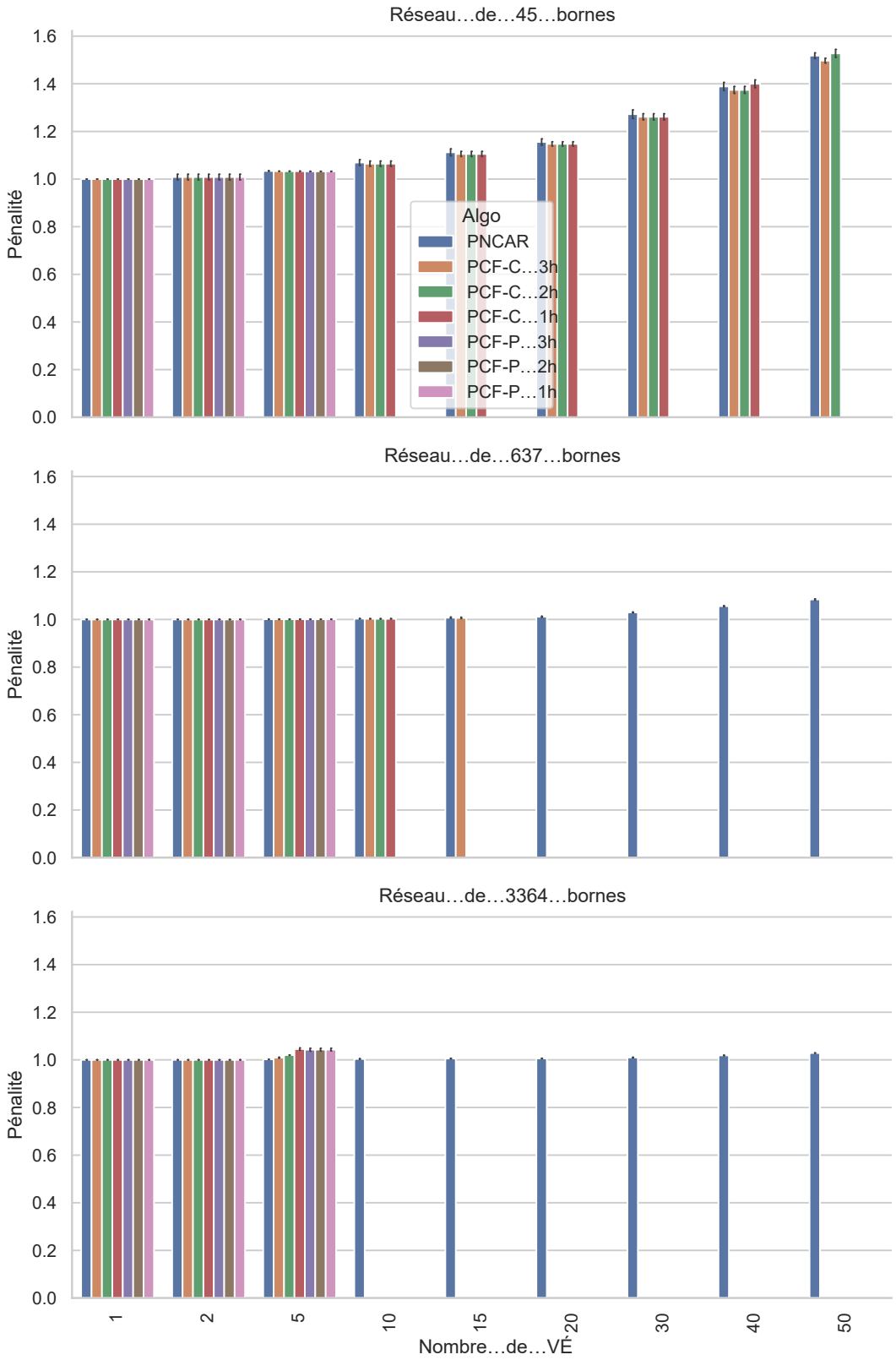


Figure 6.5 Pénalités des planificateurs en ligne fenêtrés

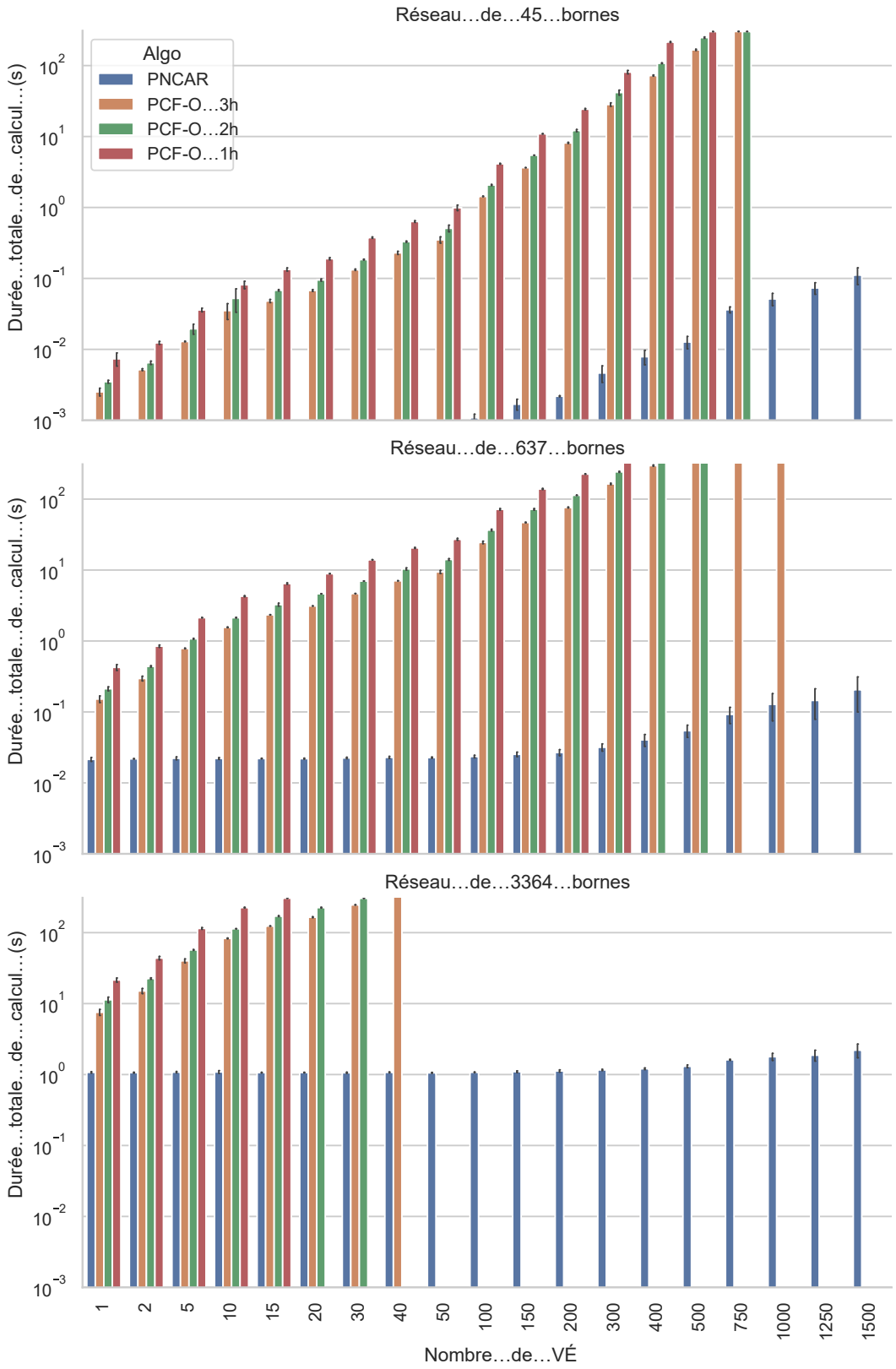


Figure 6.6 Durée des planificateurs en ligne fenêtrés

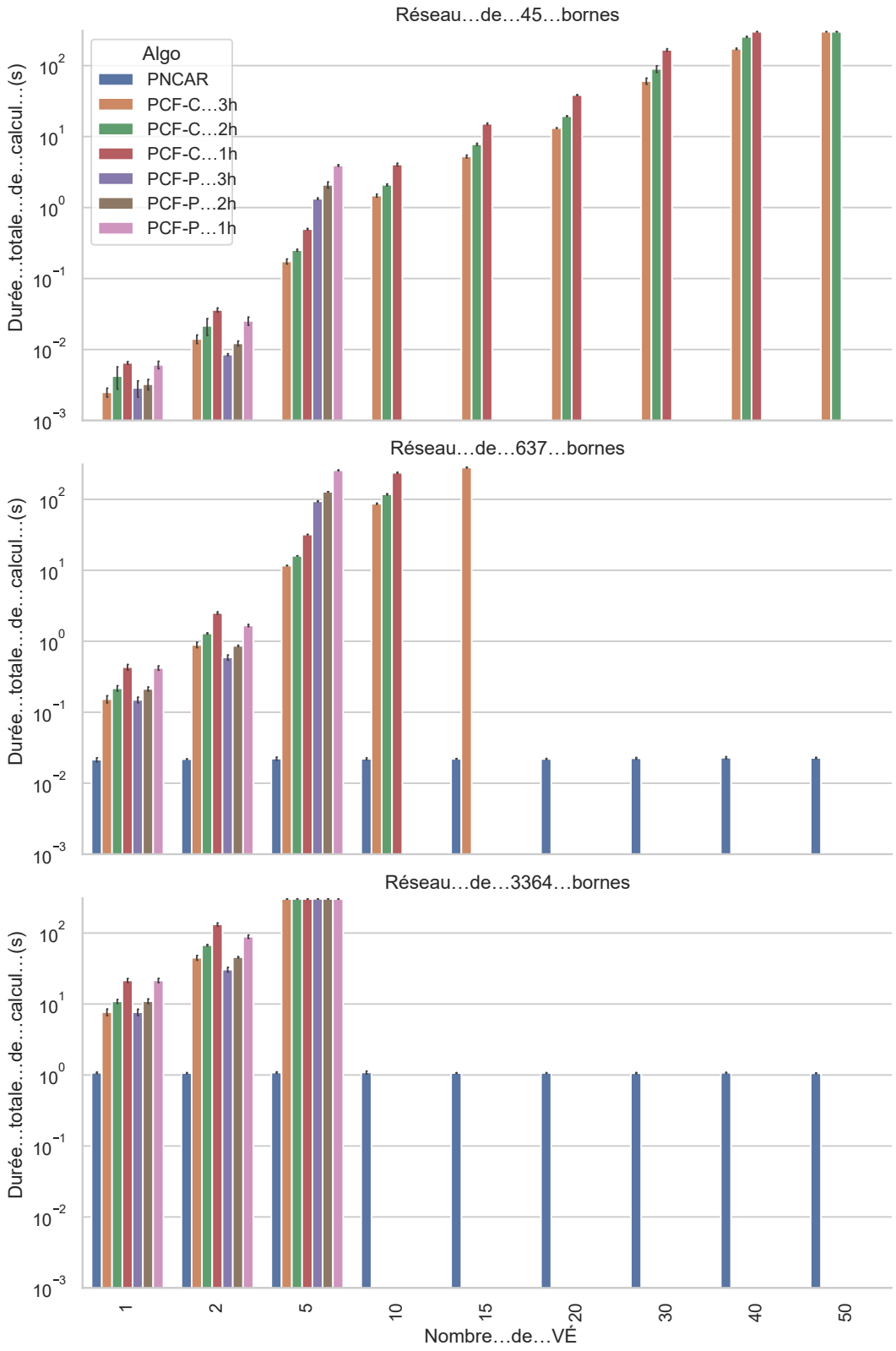


Figure 6.7 Durée des planificateurs en ligne fenêtrés

Qualité des plans générés

Les itinéraires calculés présentent des résultats généralement identiques à PNCAR. L'exception est PCF-O sur le réseau moyen et le réseau petit. Par exemple, sur le réseau moyen, la petite fenêtre présente une amélioration de 1.2 % dès que 50 VÉ sont présents sur la route et s'accentue à mesure que d'autres VÉ s'ajoutent. La fenêtre moyenne améliore la solution à 400 VÉ de 2.3 % mais aucune amélioration n'est notée pour la grande fenêtre lors de l'interruption de calcul à 1000 VÉ.

Durée de calcul

Sur le petit et moyen réseau, on remarque que PCF-O pour les 3 fenêtres étudiées, sont les plus rapides et sont interrompues après tous les variations de PCF-C et PCF-P. On observe au passage un résultat attendu : la durée de calcul augmente pour une petite fenêtre par rapport à une plus grande fenêtre.

PCF-C est impraticable : aucune amélioration n'est observée par rapport à PNCAR avant de dépasser la durée de calcul maximale. PCF-P est interrompu plus tôt que PCF-C à cause de la complexité factorielle et ne présente pas d'amélioration non plus.

Conclusion

La durée de calcul est trop importante et le problème traité ne met pas assez en évidence la différence entre les planificateurs coopératifs fenêtrés. Il n'est pas possible de tirer une conclusion convaincante. L'approche donnant quelques résultats intéressants reste PCF-O avec une petite fenêtre. Les observations discutées dans la conclusion des planificateurs hors lignes (section 6.2.1) s'appliquent également pour les planificateurs coopératifs fenêtrés.

6.3.2 Planificateurs coopératifs avec réparation locale

Rappelons que PCARL est paramétrable : trois paramètres binaires totalisant huit variations qui ont été présentées dans la section 4.7. Les pénalités sont illustrées sur 6.8 et 6.9. Les durées de calcul se trouvent sur les figures 6.10 et 6.11. Le planificateur PNCAR a été ajouté comme référence.

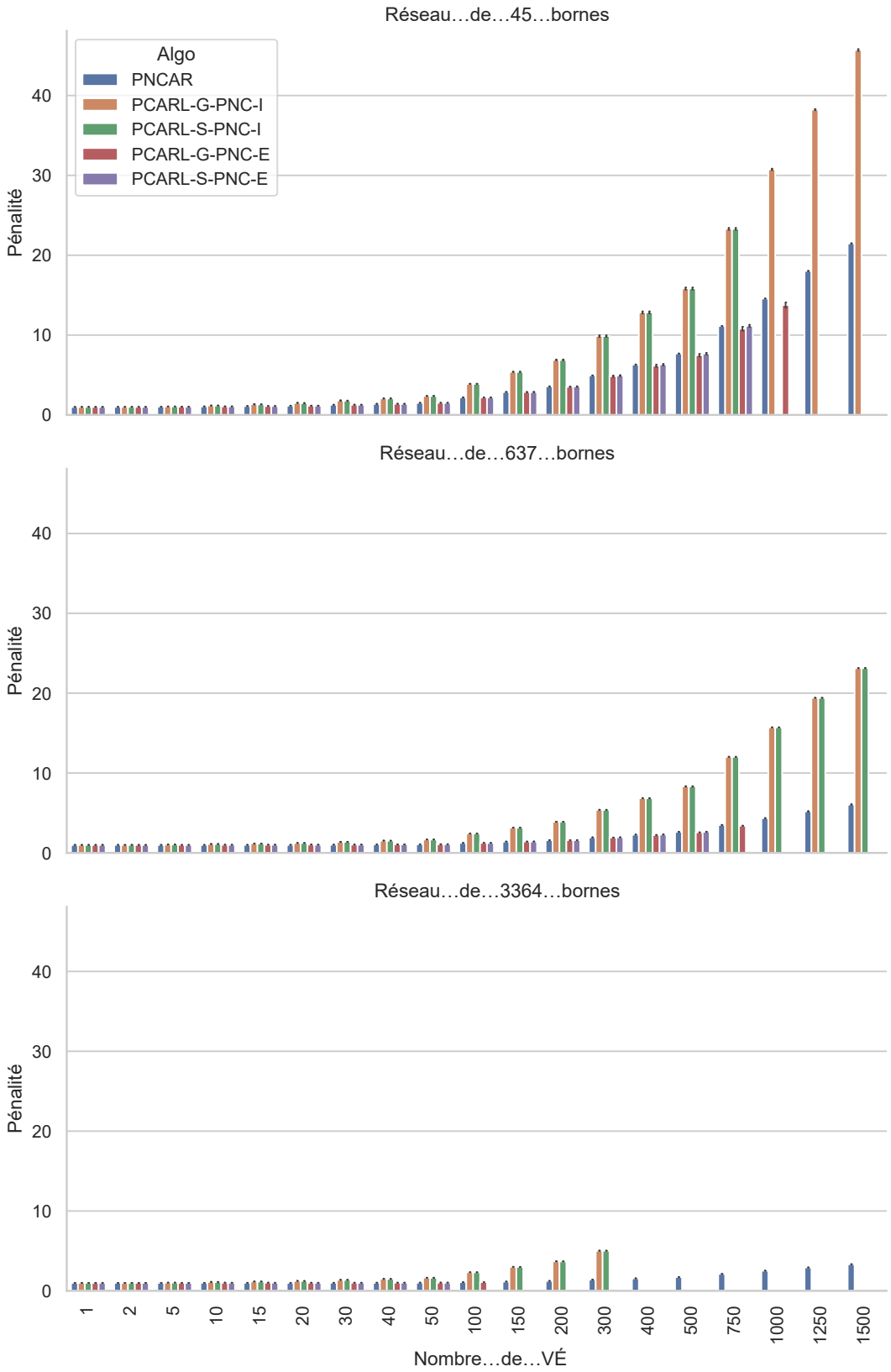


Figure 6.8 Pénalités des planificateurs avec réparation locale

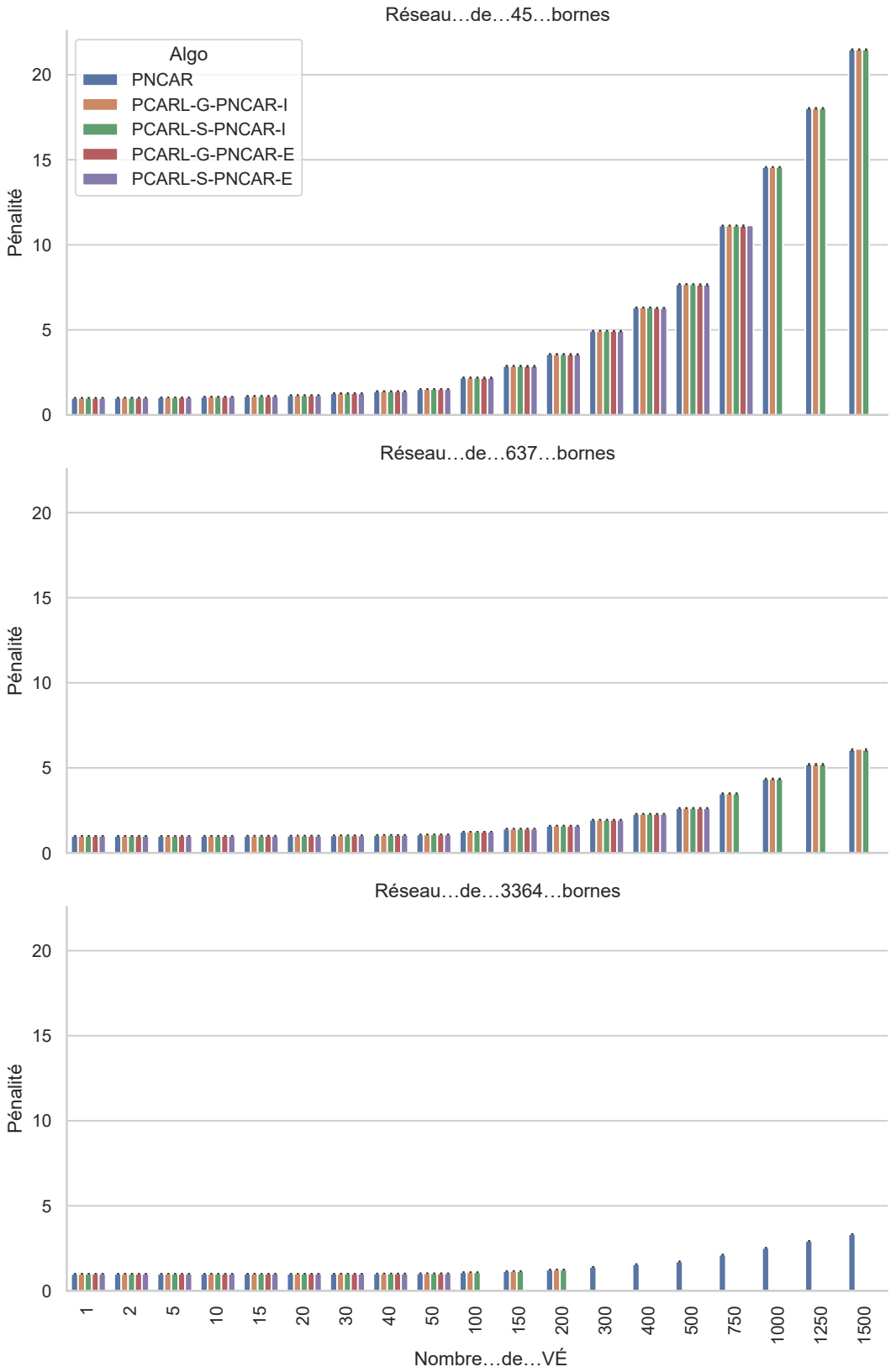


Figure 6.9 Pénalités des planificateurs avec réparation locale

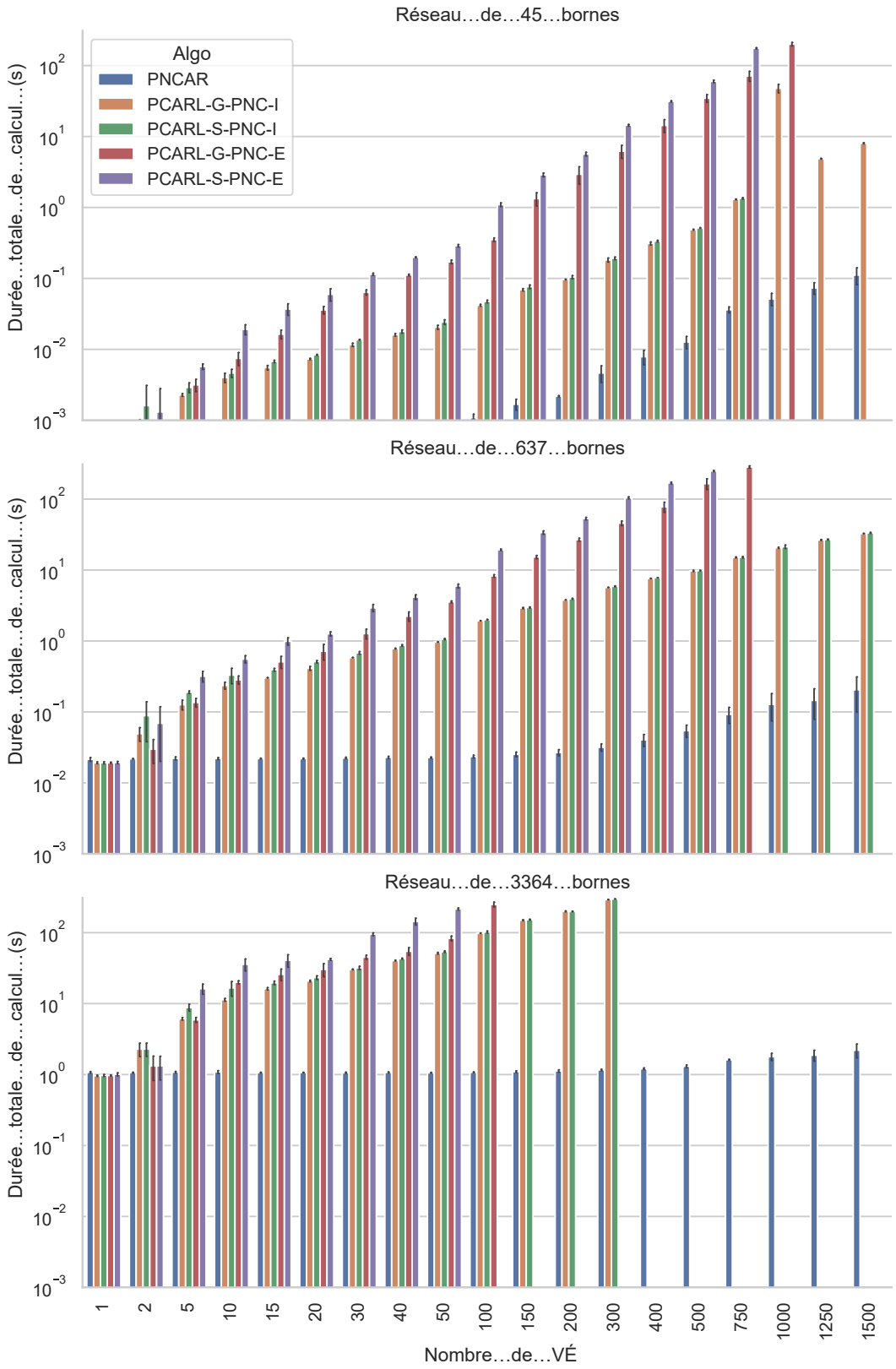


Figure 6.10 Durée de calcul des planificateurs avec réparation locale

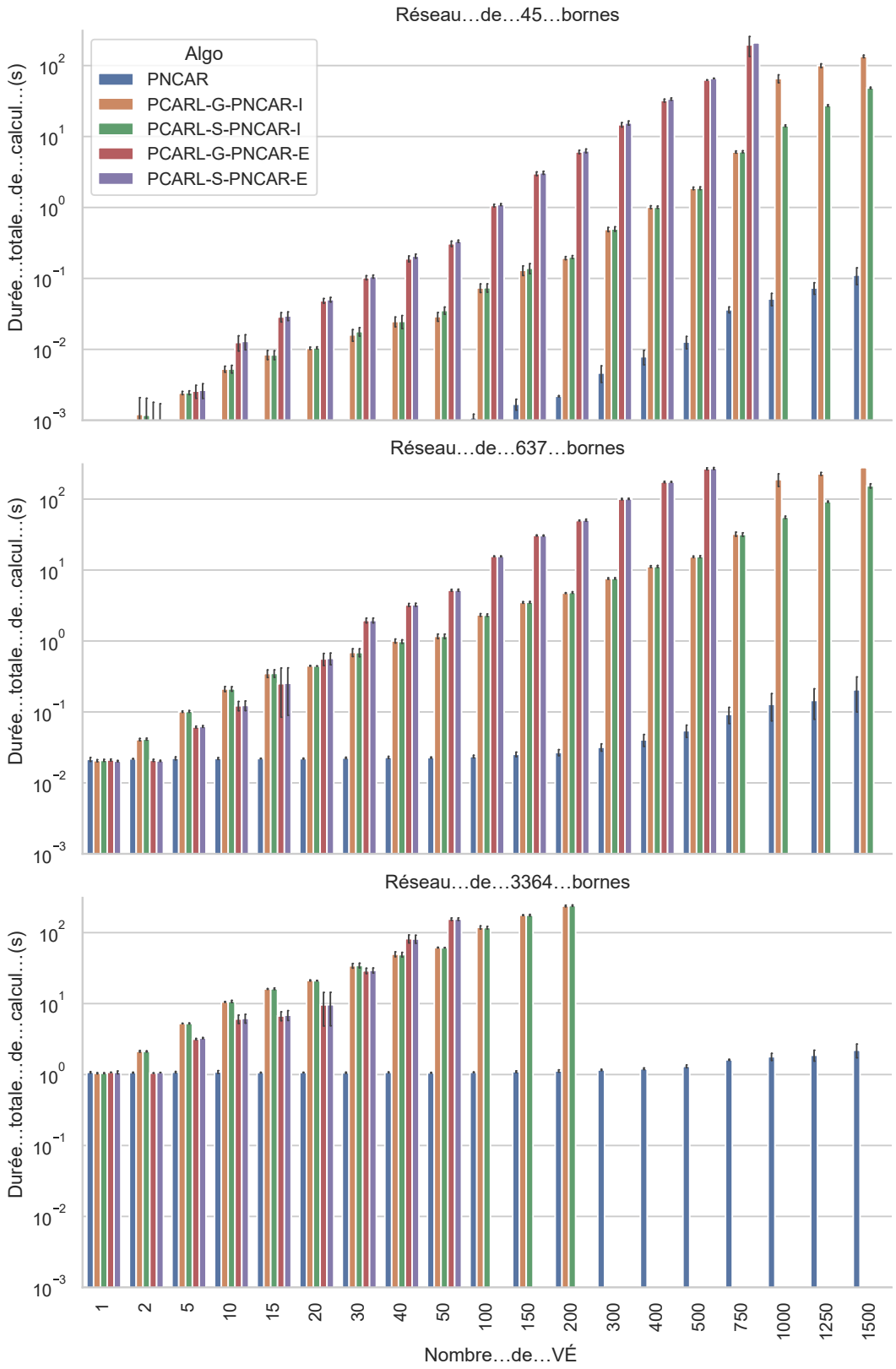


Figure 6.11 Durée de calcul des planificateurs avec réparation locale

Qualité de plans générés

Les planificateurs sont classifiés dans la table 6.1 relativement à la pénalité de PNCAR.

Pire que PNCAR	Meilleur que PNCAR	Identique à PNCAR
PCARL-G-PNC-I	PCARL-G-PNC-E	PCARL-G-PNCAR-I
PCARL-S-PNC-I		PCARL-S-PNCAR-I
		PCARL-G-PNCAR-E
		PCARL-S-PNCAR-E
		PCARL-S-PNC-E

Tableau 6.1 Classification des pénalités des planificateurs basés sur la réparation locale comparée à PNCAR.

Approches moins performantes que PNCAR

Les approches basées sur PNC qui sont incrémentales donnent de pires résultats que PNCAR : le minimum local trouvé avec cette technique est systématiquement pire que de simplement utiliser PNCAR. Ces techniques sont inintéressantes.

Approche plus performante que PNCAR

Sur le réseau moyen, PCARL-G-PNC-E présente une amélioration de 5,2% à 750 VÉ mais dépasse par la suite la durée maximale de simulation. Rappelons que l'explication des variations de PCARL se trouve à la section 4.7.

Approches aussi performantes que PNCAR

Toutes les approches basées sur PNCAR sont aussi performantes que PNCAR. Cela signifie qu'une réparation locale ne peut améliorer le résultat si la solution de départ est celle de PNCAR. Nous concluons donc expérimentalement que PNCAR est un minima local.

De manière intéressante, on remarque que PCARL-S-PNC-E présente la même pénalité que PNCAR. PCARL-S-PNC-E est la version sobre de PCARL-G-PNC-E, seule variation ayant une pénalité plus basse que PNCAR. En allouant davantage de temps de calcul à PCARL-S-PNC-E, il est possible que PCARL-S-PNC-E donne de meilleurs résultats que PNCAR et même PCARL-G-PNC-E. Cette hypothèse sera testée pour la suite du mémoire.

Durée de calcul

À nombre de VÉ égaux, les durées de calcul présentent une amélioration significative par rapport aux approches coopératives fenêtrées ou non. Les approches entières (notation -E) sont les plus longues à calculer à l'inverse incrémentales (notation -I). C'est ce qui est attendu : elles doivent recalculer l'itinéraire de tous les VÉ déjà présents sur la route. Les deux variations avec les plus petites durées de calcul sont PCARL-G-ONC-I et PCARL-G-PNCAR-I : ces approches sont incrémentales et gloutonnes.

Conclusion

L'algorithme PCARL-G-PNC-C présente à la fois une meilleure pénalité que PNF-O (1 h) et une durée de calcul plus rapide. Sur le petit réseau à 500 VÉ, PNF-O avec $\delta_{\text{fenêtre}} = 1$ h) prend 307 s à calculer pour une pénalité de 7,76 et PCARL-

G-PNC-E, 41 s pour une pénalité de 7,43.

PCARL-S-PNC-E est un planificateur à explorer davantage malgré sa pénalité équivalente à PNCAR : il faudrait simplifier le problème ou augmenter la durée maximale de calcul allouée pour tirer une conclusion.

6.3.3 Autonomie des VÉ

Depuis le début du présent chapitre, la flotte de VÉ est seulement composée du modèle Ioniq 5. Sur les huit modèles composant la distribution de part de marché du tableau 5.1, elle se classe avant-dernière en termes d'autonomie. Que se passe-t-il si on planifie sur une flotte de modèles variés suivant la distribution des parts de marché ?

La pénalité pour PNCAR, PCF-O (1 h) et PCARL-G-PNC-E est comparée sur la figure 6.12 pour ces deux profils de flotte de VÉ. La durée de calcul se trouve sur la figure 6.13.

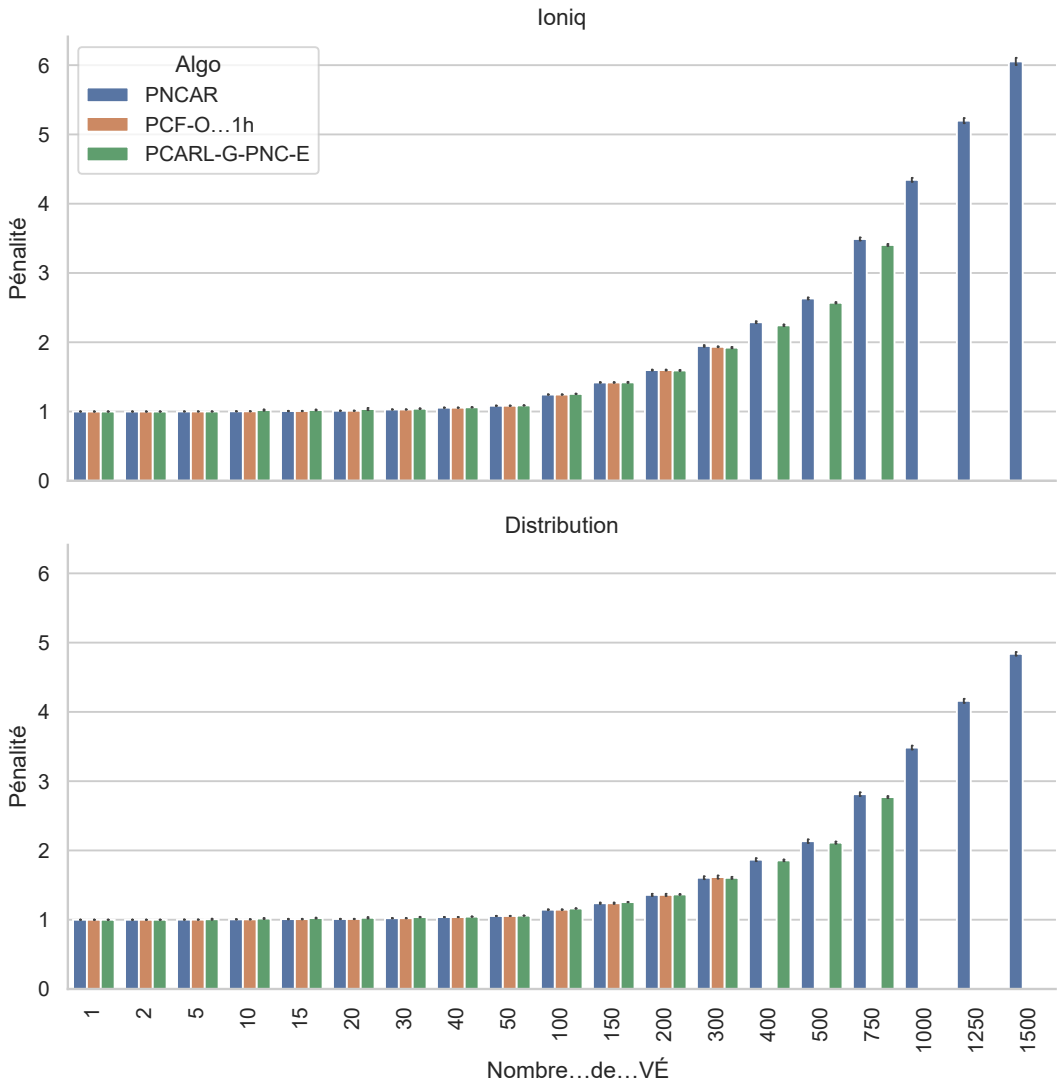


Figure 6.12 Pénalité selon l'autonomie des VÉ.

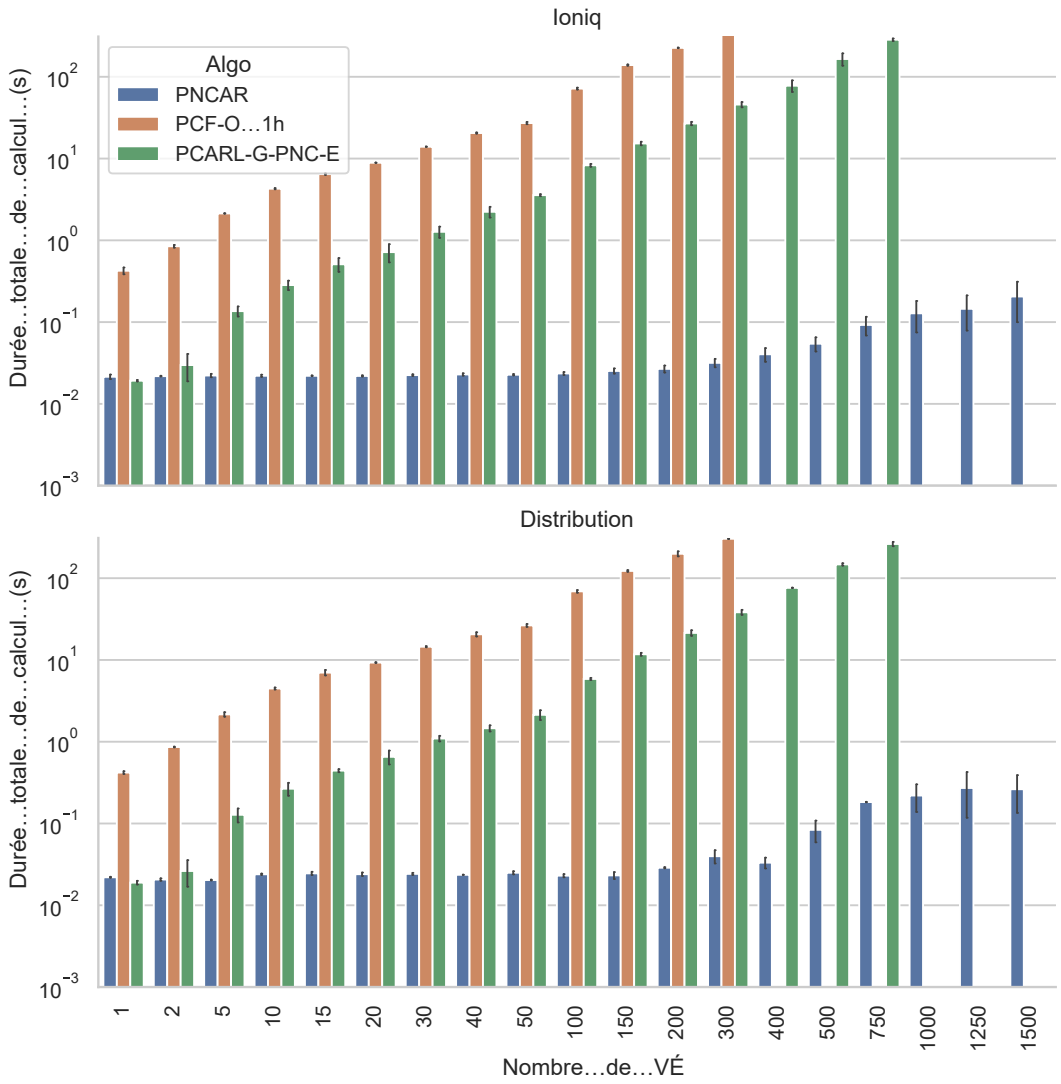


Figure 6.13 Temps de calcul selon l'autonomie des VÉ.

Qualité des plans générés

La pénalité n'augmente pas aussi rapidement pour la flotte variée que pour la flotte de Ioniq 5. C'est un résultat cohérent : la majorité des VÉ de la flotte variée ayant une autonomie supérieure, moins d'arrêts sont nécessaires : des itinéraires seront plus courts en plus de décongestionner le réseau de bornes.

L'amélioration relative à PNCAR est moins marquée pour les planificateurs coopératifs pour une flotte variée. Par exemple, pour PCF-O (1h) avec 300 VÉ, par rapport à PNCAR, la pénalité est améliorée de 2,4% pour la flotte de Ioniq 5 comparé à 0,8% pour la flotte variée.

Ces résultats confirment que le problème d'optimisation des itinéraires de VÉ sur un réseau de bornes de chargement devient moins pertinent lorsque l'autonomie augmente.

Durée de calcul

Les durées de calcul sont analogues pour les deux types de flottes. Les planificateurs coopératifs dépassent la durée maximale de calcul au même nombre de VÉ pour les deux types de flotte. La raison derrière cette observation est difficilement justifiable. D'une part, davantage d'autonomie augmente la durée de calcul, car davantage de bornes sont accessibles. En contrepartie, l'itinéraire peut contenir moins d'arrêts de chargement à effectuer, réduisant le temps de recherche.

CONCLUSION

Dans les planificateurs coopératifs hors ligne étudiés, PC-C (*planificateur coopératif - cascade*) démontre une amélioration par rapport au planificateur non coopératif PNCAR (*planificateur non coopératif avec réservation*). Sa durée de calcul augmentant dramatiquement avec le nombre de VÉ (*véhicules électriques*) limite son application. PCC (*planificateur coopératif complet*), l'approche garantie optimale, est trop longue à exécuter même pour un seul VÉ.

Pour permettre une application en ligne, deux contraintes sont ajoutées : l'insertion en continu des VÉ et la notion d'engagement. Les planificateurs coopératifs en ligne sont conçus pour respecter ces contraintes. Les plus performants sont PCF-O (1h) (*planificateur coopératif fenêtré - ordre d'arrivé*) et PCARL-G-PNC-E (*planificateur coopératif avec réparation locale gloutonne entière basée sur le planificateur non coopératif*). La pénalité s'améliore de moins de 4 % par rapport à PNCAR même pour de 500 VÉ sur le réseau moyen. Cette amélioration est relativement faible et davantage de recherche doit être effectuée.

Le réseau de borne moyen a été approfondi, car il ressemble au réseau de bornes actuellement déployé au Québec. Lorsque les planificateurs s'exécutent sur le grand réseau, l'efficacité des planificateurs est diminuée. L'efficacité des planificateurs diminue aussi lorsque l'autonomie de la flotte de VÉ augmente. Autrement dit, à mesure que le réseau de bornes se développera et que la technologie reliée aux piles de VÉ s'améliorera, le problème d'optimisation de l'itinéraire de VÉ sera moins pertinent.

Plusieurs opportunités se présentent pour poursuivre la recherche débutée dans

ce mémoire. PCARL-S-PNC-E, la version sobre de PCARL-G-PNC-E, devrait se faire allouer davantage de temps de calcul pour évaluer si une amélioration est présente. De manière plus intéressante, des heuristiques pour réduire la durée de calcul de PCC sont envisageables. Sachant que PNCAR est rapide à calculer, des approches de séparation et évaluation peuvent parallèlement être exploitées (Morrison *et al.*, 2016).

Nom de la borne	Niveau	Coordonnées en degré (latitude ;longitude)
BRCC-LesPèresNature	L3	(46.125588 ;-70.684813)
BRCC-Ste-Anne-des-Monts	L3	(49.12436 ;-66.491212)
BRCC–LeCoqRôti–L'Étape	L3	(47.560204 ;-71.23043)
BRCC-Ste-Julie	L3	(45.577444 ;-73.328066)
BRCC-VilledeMont-Joli	L3	(48.585358 ;-68.188612)
St-Hubert-Boucherville	L3	(45.600662 ;-73.447509)
BRCC-St-Hubert-Vaudreuil-Dorion	L3	(45.408302 ;-74.034054)
Artificielle0	L3	(47.578641 ;-74.947377)
Artificielle1	L3	(48.820721 ;-70.784028)
Artificielle2	L3	(46.762433 ;-72.858481)
Artificielle3	L3	(48.551555 ;-69.394142)
Artificielle4	L3	(48.239173 ;-74.531793)
Artificielle5	L3	(48.131125 ;-71.069509)
Artificielle6	L3	(47.953073 ;-72.680529)
Artificielle7	L3	(48.972348 ;-72.703146)
Artificielle8	L3	(48.923296 ;-69.655393)
Artificielle9	L3	(45.652095 ;-76.124974)

Tableau 6.2 Échantillon d'un fichier CSV de bornes. 6 bornes sont exportées de OSM et 10 bornes sont générées.

BorneDépart	BorneArrivée	Distance(m)
0	8	271697
0	31	87635.4
0	34	60147.6
0	37	51143
0	38	92043.9
0	39	162227
0	40	213359
0	41	290094
0	42	346853
1	2	54839.1
1	3	106419
1	4	146700
1	5	151625
1	6	140374
1	7	101272
1	8	218831

Tableau 6.3 Exemple de graphe des bornes. Chaque entrée est une paire de bornes entre la destination et l'arrivée accompagnée de la distance. La borne 0 est liée à 9 autres bornes, les autres dépassant 283000 m.

Nom	Coordonnée haut-gauche (latitude,longitude)	Coordonnée bas droite (latitude,longitude)	Description
Montreal	(45.567684, -73.560965)	(45.475795, -73.656416)	Englobe complètement Montréal et une partie du grand Montréal.
Matane	(48.978627, -66.975120)	(48.390472, -67.632531)	Englobe Matane et Trois-Pistol.
Saguenay	(48.929216, -72.503957)	(48.292208, -70.854881)	Englobe Saguenay et Alma.
Mégantic	(45.711067, -71.692751)	(45.462250, -70.741759)	Englobe Sherbrooke, Lennoxville et Magog.

Tableau 6.4 Liste des zones de départs et arrivées accompagnée de la position du coin supérieur gauche et du coin inférieur droit.

Nom véhicule	Coordonnée départ (latitude;longitude)	Coordonné arrivé (latitude ;longitude)	Autonomie(m)	Temps départ(s)	Temps charge(s)
Vehicle1	(45.484200509634057141 ;-73.6133943151067171 58)	(48.687126109342166558 ;-67.5785340351323924 32)	250000	345	1800
Vehicle11	(48.469056413229075986 ;-67.468170363195667 96)	(45.538874341012792968 ;-73.635822390202037013)	250000	345	1800
Vehicle8	(45.499124263052635797 ;-73.604197534081293952)	(48.744629924145705502 ;-67.316087901179088249)	250000	402	1800
Vehicle18	(48.559064591547851819 ;-67.141980221677357576)	(45.533547097517633517 ;-73.618581907507746109)	250000	402	1800
Vehicle9	(45.543970024123190399 ;-73.571100578891588384)	(48.816332682585915848 ;-67.408994288352175772)	250000	1428	1800
Vehicle19	(48.686405862576719983 ;-67.429212420597565369)	(45.534202307709982449 ;-73.647765600013457288)	250000	1428	1800
Vehicle4	(45.552464174863018798 ;-73.650505861516819 55)	(48.957544431737069601 ;-67.517075475297659182)	250000	1522	1800
Vehicle14	(48.475840158747899977 ;-67.421670717398569 65)	(45.551829074392102825 ;-73.635838772425969978)	250000	1522	1800
Vehicle10	(45.562065414985255529 ;-73.603916731479429814)	(48.403676286974416598 ;-67.097808623330578826)	250000	1721	1800
Vehicle20	(48.719772082842936811 ;-67.466535866341359906)	(45.522002701917408274 ;-73.634677465017631448)	250000	1721	1800

Tableau 6.5 Échantillon d'une requête entre Montréal et Matane. Dix des 1500 requêtes sont présentés ici.

Nom :	VE_1365					
Temps de départ :	16s					
Total :	43960s					
Attente :	0					
Charge :	11041s					
Route :	32919s					
—CHEMIN—						
État	Noeud de départ	(Longitude, latitude)	Départ(s)		Arrivée(s)	Durée(s)
Conduite	Point de départ	(48.9318,-67.441)	24	-	24	0
Conduite	to charger 38	(48.4273,-68.5114)	24	-	5143	5119
Charge	from charger 38	(48.4273,-68.5114)	5143	-	7190	2047
Conduite	to charger 40	(47.7643,-69.5914)	7190	-	13306	6116
Charge	from charger 40	(47.7643,-69.5914)	13306	-	15752	2446
Conduite	to charger 41	(47.2392,-70.2121)	15752	-	19592	3840
Charge	from charger 41	(47.2392,-70.2121)	19592	-	21128	1536
Conduite	to charger 42	(46.9269,-70.6735)	21128	-	23965	2837
Charge	from charger 42	(46.9269,-70.6735)	23965	-	25100	1135
Conduite	to charger 43	(46.4161,-71.7996)	25100	-	30647	5547
Charge	from charger 43	(46.4161,-71.7996)	30647	-	32866	2219
Conduite	to charger 2	(45.8989,-72.5257)	32866	-	37011	4145
Charge	from charger 2	(45.8989,-72.5257)	37011	-	38669	1658
Conduite	to destination	(45.5195,-73.6504)	38669	-	43984	5315

Tableau 6.6 Exemple de plan local pour un VÉ nommé VE_1365 utisant PI.

Nombre de VE: 10

**********Methode**********			*****Temps moy.trajet(s)*****				***Temps calcul(s)****			****Metrique****
Nom	Combinaison	Parametre	Total	= Route	+ Att.	+ Charge	Total	= Pass *	Temps/p	Penalite
Cache			420	327	0	93	2.011	1	2.011	1.000
PNCSR			420	327	0	93	0.000	1	0.000	1.000
PNCAR			545	327	125	93	0.000	1	0.000	1.304
PNCARR			465	344	20	100	0.000	1	0.000	1.110
PCAPDPPremier			465	344	20	100	0.005	1	0.005	1.110
PCAPF	Premier	W=10800 R=21600	465	344	20	100	0.037	40	0.001	1.110
PCAPF	Premier	W=7200 R=14400	465	344	20	100	0.050	60	0.001	1.110
PCAPF	Premier	W=3600 R=7200	465	344	20	100	0.098	116	0.001	1.110
PCAPDPCascade			459	341	20	99	0.489	1	0.489	1.096
PCAPF	Cascade	W=10800 R=21600	459	341	20	99	3.746	40	0.094	1.096
PCAPF	Cascade	W=7200 R=14400	459	341	20	99	5.678	60	0.095	1.096
PCAPF	Cascade	W=3600 R=7200	459	341	20	99	11.145	117	0.095	1.096
PCAPDPPermutation			149	112	1	36	0.139	1	0.139	1.027
PCAPF	Permutation	W=10800 R=21600	149	112	1	36	1.231	25	0.049	1.027
PCAPF	Permutation	W=7200 R=14400	149	112	1	36	1.769	37	0.048	1.027
PCAPF	Permutation	W=3600 R=7200	149	112	1	36	3.399	72	0.047	1.027
PCARLG_PNCARR			465	344	20	100	0.007	1	0.007	1.110
PCARL_PNCARR			465	344	20	100	0.007	1	0.007	1.110
PCARLG_PNCARR_C			465	344	20	100	0.018	1	0.018	1.110
PCARL_PNCARR_C			465	344	20	100	0.019	1	0.019	1.110
PCARLG_PNCAR			529	328	106	94	0.004	1	0.004	1.263
PCARL_PNCAR			529	328	106	94	0.005	1	0.005	1.263
PCARLG_PNCAR_C			474	342	31	100	0.011	1	0.011	1.132
PCARL_PNCAR_C			465	344	21	100	0.022	1	0.022	1.110
PCC			0	0	0	0	0.000	0	-nan(ind)	-nan(ind)

Figure 6.14 Sommaire de la performance des planificateurs coopératifs pour une situation à 10 VÉ sur un petit réseau.

NbBorne	NbVoiture	Algo	FO	Metrique	TotMin	TotMax	TotalMoyen	RouteMoyenne	AttenteMoyenne	ChargeMoyenne	TotalEcartType	RouteEcartType	AttenteEcartType	ChargeEcartType	TempsCalcul	NP	Autonomie
45	10	Cache	Penalite	1	39216	46082	42317.5	32773.89844	0	9543.599609	1190.440063	690.416931	0	1457.692993	8.173599	1	Distribution
45	10	PNCSR	Penalite	1	39216	46082	42317.5	32773.89844	0	9543.599609	1190.440063	690.416931	0	1457.692993	0.000472	1	Distribution
45	10	PNCAR	Penalite	1.214716	43749	61764	51161.60156	32773.89844	8844.099609	9543.599609	1606.309814	690.416931	6763.035645	1457.692993	0.000385	1	Distribution
45	10	PNCARR	Penalite	1.078697	43292	47649	45546.90234	34019.60156	1367	10160.29981	169.213043	1134.026978	1130.401367	946.416321	0.000411	1	Distribution
45	10	PCAPDPPremier	Penalite	1.078697	43292	47649	45546.90234	34019.60156	1367	10160.29981	169.213043	1134.026978	1130.401367	946.416321	0.003252	1	Distribution
45	10	PCAPFPremier_large	Penalite	1.078697	43292	47649	45546.90234	34019.60156	1367	10160.29981	169.213043	1134.026978	1130.401367	946.416321	0.023235	32	Distribution
45	10	PCAPFPremier_medium	Penalite	1.078697	43292	47649	45546.90234	34019.60156	1367	10160.29981	169.213043	1134.026978	1130.401367	946.416321	0.032706	48	Distribution
45	10	PCAPFPremier_small	Penalite	1.078697	43292	47649	45546.90234	34019.60156	1367	10160.29981	169.213043	1134.026978	1130.401367	946.416321	0.060664	92	Distribution
45	10	PCAPDPCascade	Penalite	1.064955	42651	46509	44989.59766	33726.89844	1219.400024	10043.29981	345.447754	876.190308	1156.697998	1146.188965	0.147308	1	Distribution
45	10	PCAPFCascade_large	Penalite	1.064955	42651	46509	44989.59766	33726.89844	1219.400024	10043.29981	345.447754	876.190308	1156.697998	1146.188965	1.127051	32	Distribution
45	10	PCAPFCascade_medium	Penalite	1.064955	42651	46509	44989.59766	33726.89844	1219.400024	10043.29981	345.447754	876.190308	1156.697998	1146.188965	1.662021	48	Distribution
45	10	PCAPFCascade_small	Penalite	1.064955	42651	46509	44989.59766	33726.89844	1219.400024	10043.29981	345.447754	876.190308	1156.697998	1146.188965	3.16783	93	Distribution
45	10	PCARLG_PNCARR	Penalite	1.078697	43292	47649	45546.90234	34019.60156	1367	10160.29981	169.213043	1134.026978	1130.401367	946.416321	0.003022	1	Distribution
45	10	PCARL_PNCARR	Penalite	1.078697	43292	47649	45546.90234	34019.60156	1367	10160.29981	169.213043	1134.026978	1130.401367	946.416321	0.003048	1	Distribution
45	10	PCARLG_PNCARR_C	Penalite	1.078697	43292	47649	45546.90234	34019.60156	1367	10160.29981	169.213043	1134.026978	1130.401367	946.416321	0.008925	1	Distribution
45	10	PCARL_PNCARR_C	Penalite	1.078697	43292	47649	45546.90234	34019.60156	1367	10160.29981	169.213043	1134.026978	1130.401367	946.416321	0.009585	1	Distribution
45	10	PCARLG_PNCAR	Penalite	1.196096	43749	61764	50365.69922	32839	7939.299805	9587.400391	1354.625	751.982971	7262.068359	1464.692261	0.003183	1	Distribution
45	10	PCARL_PNCAR	Penalite	1.196096	43749	61764	50365.69922	32839	7939.299805	9587.400391	1354.625	751.982971	7262.068359	1464.692261	0.003611	1	Distribution
45	10	PCARLG_PNCAR_C	Penalite	1.078697	43292	47649	45546.89844	33986.5	1413.199951	10147.2002	169.213425	1145.793579	1195.968872	947.915649	0.00792	1	Distribution
45	10	PCARL_PNCAR_C	Penalite	1.078697	43292	47649	45546.89844	33986.5	1413.199951	10147.2002	169.213425	1145.793579	1195.968872	947.915649	0.012309	1	Distribution

Tableau 6.7 Données exportées pour une expérience d'un réseau de 45 bornes comprenant dix VÉ en utilisant la fonction objective de pénalité.

RÉFÉRENCES

Abousleiman, R. et Rawashdeh, O. (2015). Energy Consumption Model of an Electric Vehicle. *IEEE Transportation Electrification Conference and Expo (ITEC)*. http://dx.doi.org/10.1109/ITEC.2015.7165773

Association des Véhicules Électriques du Québec (2021). Statistiques SAAQ-AVÉQ sur l'électromobilité au Québec en date du 30 juin 2021. Récupéré de https://www.aveq.ca/actualiteacutes

Automobile Propre (2021). Simulateur de Temps de Recharge Chevrolet Bolt. Récupéré de https://www.automobile-propre.com/simulateur-temps-de-recharge-voiture-electrique/chevrolet-bolt/

Beaudry, E., Brosseau, Y., Coté, C., Raevsky, C., Létourneau, D., Kabanza, F. et Michaud, F. (2005). Reactive Planning in a Motivated Behavioral Architecture. *Proceedings of the 20th National Conference on Artificial intelligence (AAAI)*, *3*, 1242–1247.

Benaskeur, A. R., Kabanza, F. et Beaudry, E. (2010). CORALS : A real-time planner for anti-air defense operations. *ACM Transactions on Intelligent Systems and Technology*, *1*(2). http://dx.doi.org/10.1145/1869397.1869402

Cao, Y., Wang, T., Kaiwartya, O., Min, G., Ahmad, N. et Abdullah, A. H. (2018). An EV Charging Management System Concerning Drivers' Trip Duration and Mobility Uncertainty. *IEEE Transactions on Systems, Man, and Cybernetics : Systems*, *48*(4), 596–607. http://dx.doi.org/10.1109/TSMC.2016.2613600

ChargeHub (2021). Trouvez toutes les bornes publiques de recharge pour VÉ. Récupéré de https://chargehub.com/fr/carte-borne-de-recharge.html#

De Nunzio, G., Ben Gharbia, I. et Sciarretta, A. (2020). A Time- And Energy-Optimal Routing Strategy for Electric Vehicles with Charging Constraints. Dans *IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*. Institute of Electrical and Electronics Engineers Inc. http://dx.doi.org/10.1109/ITSC45102.2020.9294622

Données Québec (2022). Débit de circulation - Jeu de données - Données Québec. Récupéré de https://www.donneesquebec.ca/recherche/dataset/debit-de-circulation

Eisner, J., Funke, S. et Storandt, S. (2011). Optimal Route Planning for Electric Vehicles in Large Networks. Dans *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, volume 2, 1108–1113.

Énergie et ressources naturelles Québec (2021). La recharge publique. Récupéré de https://vehiculeselectriques.gouv.qc.ca/decouvrir/recharge/recharge-publique.asp

Fiori, C., Ahn, K. et Rakha, H. A. (2016). Power-Based Electric Vehicle Energy Consumption Model : Model Development and Validation. *Applied Energy*, *168*, 257–268. http://dx.doi.org/10.1016/J.APENERGY.2016.01.097

Franke, T. et Krems, J. F. (2013). Understanding Charging Behaviour of Electric Vehicle Users. *Transportation Research Part F : Traffic Psychology and Behaviour*, *21*, 75–89. http://dx.doi.org/10.1016/J.TRF.2013.09.002

Gareau, J. C., Beaudry, et Makarenkov, V. (2019). An Efficient Electric Vehicle Path-Planner that Considers the Waiting Time. Dans *Proceedings of the ACM International Symposium on Advances in Geographic Information Systems (GIS)*, 389–397. Association for Computing Machinery. http://dx.doi.org/10.1145/3347146.3359064

Geisberger, R., Sanders, P., Schultes, D. et Vetter, C. (2012). Exact routing in large road networks using contraction hierarchies. *Transportation Science*, *46*(3), 388–404. http://dx.doi.org/10.1287/trsc.1110.0401. Récupéré de https://pubsonline.informs.org/doi/abs/10.1287/trsc.1110.0401

Ghallab, M., Nau, D. et Traverso, P. (2016). *Automated Planning and Acting*. Cambridge : Cambridge University Press. http://dx.doi.org/DOI:10.1017/CBO9781139583923

Hart, P. E., Nilsson, N. J. et Raphael, B. (1968). A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics*, *4*(2), 100–107. http://dx.doi.org/10.1109/TSSC.1968.300136

Hoke, A., Brissette, A., Maksimović, D., Pratt, A. et Smith, K. (2011). Electric Vehicle Charge Optimization Including Effects of Lithium-Ion

Battery Degradation. *IEEE Vehicle Power and Propulsion Conference (VPPC)*. http://dx.doi.org/10.1109/VPPC.2011.6043046

Hovenburg, A. R., De Alcantara Andrade, F. A., Rodin, C. D., Johansen, T. A. et Storvold, R. (2017). Contingency Path Planning for Hybrid-Electric UAS. Dans *Workshop on Research, Education and Development of Unmanned Aerial Systems, RED-UAS)*, 37–42. Institute of Electrical and Electronics Engineers Inc. http://dx.doi.org/10.1109/RED-UAS.2017.8101640

Jaël Gareau, Éric Beaudry et Vladimir Makarenkov (2022). Veplan du Laboratoire CRIA - application pour planifier son déplacement en véhicule électrique. Récupéré de http://veplan.uqam.ca/

Karloff, H. (2009). The Simplex Algorithm. In *Linear Programming* 23–47. Boston, MA : Birkhäuser Boston

Land, A. H. et Doig, A. G. (1960). An Automatic Method of Solving Discrete Programming Problems. *Econometrica*, *28*(3), 497. http://dx.doi.org/10.2307/1910129

Latombe, J.-C. (1991). *Robot Motion Planning*. Boston, MA : Springer, Boston, MA. http://dx.doi.org/10.1007/978-1-4615-4022-9{_}1

Le Circuit Électrique (2021). Trouver Une Borne du Circuit Électrique. Récupéré de https://lecircuitelectrique.com/fr/trouver-une-borne/

Lejeune, E. et Sarkar, S. (2021). Survey of the Multi-Agent Pathfinding Solutions. http://dx.doi.org/10.13140/RG.2.2.14030.28486

Lucas, A., Barranco, R. et Refa, N. (2019). EV Idle Time Estimation on Charging Infrastructure, Comparing Supervised Machine Learning Regressions. *Energies*, *12*(2), 269. http://dx.doi.org/10.3390/en12020269

Morrison, D. R., Jacobson, S. H., Sauppe, J. J. et Sewell, E. C. (2016). Branch-and-bound algorithms : A survey of recent advances in searching, branching, and pruning. *Discrete Optimization*, *19*, 79–102. http://dx.doi.org/10.1016/j.disopt.2016.01.005

Norvig Peter et Russell, S. J. S. J. (2020). *Artificial intelligence : a modern approach*. Fourth edition. Upper Saddle River, N.J. : Prentice Hall, ©2010.

OpenStreetMap Foundation (2022). OpenStreetMap. Récupéré de https://www.openstreetmap.org/#map=3/71.34/-96.82

Sachenbacher, M., Leucker, M., Artmeier, A. et Haselmayr, J. (2011). Efficient Energy-Optimal Routing for Electric Vehicles. Dans *Proceedings of*

the National Conference on Artificial Intelligence, volume 2, 1402–1407.

Sharon, G., Stern, R., Felner, A. et Sturtevant, N. R. (2015). Conflict-Based Search for Optimal Multi-Agent Pathfinding. *Artificial Intelligence*, *219*, 40–66. http://dx.doi.org/10.1016/j.artint.2014.11.006

Shi, R., Steenkiste, P. et Veloso, M. M. (2019). SC-M* : A Multi-Agent Path Planning Algorithm with Soft-Collision Constraint on Allocation of Common Resources. *Applied Sciences*, *9*(19). http://dx.doi.org/10.3390/app9194037

Silver, D. (2005). Cooperative Pathfinding. Dans *Proceedings of the 1st Artificial Intelligence and Interactive Digital Entertainment Conference (AIIDE)*, 117–122.

Société d'Assurance Automobile du Québec et Association Véhicules Électrique du Québec (2021). Statistiques SAAQ-AVÉQ sur l'électromobilité au Québec en date du 30 juin 2021 [Infographie]. Récupéré de https://www.aveq.ca/actualiteacutes

Stern, R. (2019). Multi-agent path finding – an overview. Dans Osipov Gennady S., , A. I. Panov, S, et Y. Konstantin (dir.). *Lecture Notes in Computer Science*, volume 11866 LNAI, 96–115., Cham. Springer International Publishing. http://dx.doi.org/10.1007/978-3-030-33274-7{_}6

Surynek, P. (2021). Compilation-based Solvers for Multi-Agent Path Finding : a Survey, Discussion, and Future Opportunities. Récupéré de https://arxiv.org/abs/2104.11809v1

Wagner, G. et Choset, H. (2013). M* : A Complete Multirobot Path Planning Algorithm with Optimality Bounds. Dans *Lecture Notes in Electrical Engineering (LNEE)*, volume 57, 167–181. http://dx.doi.org/10.1007/978-3-642-33971-4{_}10

Yang, S. C., Li, M., Lin, Y. et Tang, T. Q. (2014). Electric Vehicle's Electricity Consumption on a Road With Different Slope. *Physica A : Statistical Mechanics and its Applications*, *402*, 41–48. http://dx.doi.org/10.1016/j.physa.2014.01.062

Zelinsky, A. (1992). A Mobile Robot Exploration Algorithm. *IEEE Transactions on Robotics and Automation*, *8*(6), 707–717. http://dx.doi.org/10.1109/70.182671

Zilberstein, S. (1996). Using anytime algorithms in intelligent systems. *AI*

Magazine, *17*(3), 73–83.