Rapport du projet EP Tender

Maël Laoufi et Sylvain Chabredier

Nous tenons à remercier Hugo Basset et EP Tender pour nous avoir donné l'opportunité de travailler sur ce sujet, et Bertrand Maury ainsi que Louis-Pierre Chaintron pour leur encadrement et leurs propositions de pistes de réflexion.

Synthèse

Ce rapport s'intéresse au problème de rangement optimal en files de batteries électriques dans une station de recharge. Ce problème est rencontré par la start-up EP Tender dans le cadre d'un projet de location de batteries amovibles visant à augmenter l'autonomie de voitures électriques. Pour des longs trajets, les clients viendraient dans des stations relais récupérer une batterie entièrement chargée, puis à l'issue de leur voyage la rendraient dans une autre station-relais. Les batteries, ou tenders, sont rangées en station en différentes files, chacune reliée à une borne de chargement. Ces files fonctionnent selon le principe "first in, last out" : les premières batteries à arriver dans une file voient leur sortie bloquée par les batteries plus récemment arrivées. Réarranger les batteries demande de l'espace en station, consomme de l'énergie et force les clients à attendre que le réarrangement soit terminé avant d'être servi, c'est donc une pratique à éviter. Pour autant, les tenders sont coûteux et l'entreprise souhaite donc maximiser leur utilisation. Le problème s'est donc posé à nous comme suit : **comment ranger les tenders en station de manière à minimiser le temps d'attente des clients, tout en évitant de réarranger les batteries entre elles ?**

Trois étapes se succèdent dans ce rapport.

Tout d'abord, l'introduction de modèles déterministes où les clients arrivent de manière parfaitement régulière nous permet d'appréhender le rôle des différents paramètres des stations. La puissance de charge des files et le nombre total de files permettent de jouer sur la puissance de charge de la station. Plus cette dernière est élevée, et plus on peut absorber un flux élevé de clients. Le stock de tenders en station n'a pas d'effet sur le long terme mais un stock élevé permet de mieux absorber une augmentation temporaire du flux de clients. On introduit l'algorithme glouton, qui consiste à ranger les tenders déchargés dans les files les moins chargées, et on montre qu'il fait partie des algorithmes optimaux dans ce cadre : il permet d'absorber sur le long terme le flux maximal de clients.

Dans la suite, on utilise des modèles aléatoires afin d'évaluer la robustesse de l'algorithme glouton à un flux de clients qui n'est pas parfaitement connu. On y montre que si l'algorithme glouton reste optimal dans les cas limites où on a très peu ou beaucoup de clients, dans les cas plus réalistes avec un flux de clients modéré, ce n'est pas toujours le cas. On démontre l'apparition d'un compromis longueur/charge : il est parfois utile de ne pas ranger les tenders déchargés dans la file la moins chargée si cette dernière est longue, mais plutôt dans des files plus chargées mais plus courtes, afin de laisser un plus grand nombre de tenders se charger complètement. L'étude des modèles met en évidence que, pour des valeurs qui restent modérées, le compromis longueur/charge croît en importance avec l'intensité de la demande. Ainsi, en heure de pointe, il pourra être utile de s'écarter légèrement de l'algorithme glouton pour favoriser le rangement des tenders dans des files plus courtes.

Pour finir, on conduit une analyse exploratoire de différentes classes d'algorithmes permettant d'implémenter en pratique ce compromis longueur/charge. Pour des modèles réalistes, une fois ces algorithmes bien calibrés, ils permettent en effet de soutenir la demande des clients sans qu'il n'y ait aucun temps d'attente sur une période plus longue. Cependant, ce gain est minime, de l'ordre d'un à deux pourcents.

En conclusion, l'algorithme glouton est un bon algorithme pour le rangement des tenders. Son comportement est prévisible, et il se comporte de manière quasi-optimale indépendamment de la taille de la station ou du flux de clients. Cependant, dans les phases d'activité plus intense, ou dans l'éventualité d'une croissance rapide d'EP Tender conduisant à un flux de client trop intense pour les stations, un gain d'efficacité peut être réalisé en implémentant une des classes d'algorithmes proposées, par exemple celle basée sur les indices multiplicatifs. Le paramètre variable devra être calibré à la taille des stations et à la fréquence estimée d'arrivée des clients, qui peut varier selon le moment de la journée. Le gain en résilience des stations ainsi réalisé sera néanmoins minime.

Table des matières

1 Introduction

1.1 Présentation du problème

Comme mentionné dans la synthèse, le problème auquel on s'intéresse dans ce rapport est un problème d'arrangement optimal en files de batteries dans une station. Précisons ici un peu la situation.

Le principe est le suivant : on dispose d'une station qui accueille des tenders. Cette station dispose d'un nombre f de files fixé, et tous les tenders sont arrangés dans ces files. Ces files fonctionnent selon le principe "first in, last out". Les tenders ont un niveau de charge, et lorsqu'ils sont connectés à une file, il peuvent se recharger, selon la capacité de charge de la file. Des voitures arrivent devant la station : elles peuvent demander un tender pour augmenter leur autonomie, rendre un tender qu'elles avaient emprunté, ou les deux à la fois. On s'impose les contraintes suivantes :

1. les tenders qui sont déjà dans la station ne peuvent être réarrangés,
2. on ne peut prendre dans chaque file que le dernier tender arrivé,
3. lorsqu'une voiture demande un tender, il faut lui en donner un pleinement chargé.

Ce rapport traite de la question de l'arrangement des tenders : il essaie de déterminer comment ranger les tenders qui reviennent déchargés et comment choisir quels tenders chargés donner aux clients. Pour minimiser le mécontentement des clients, on essaie de trouver l'algorithme optimal qui permette de faire attendre le moins possible les voitures clientes lorsqu'elles demandent un tender.

Mentionnons qu'on a déjà l'intuition du problème : les tenders qui sont dans les premières places d'une file sont ceux qui sont dans la station depuis le plus longtemps, donc c'est a priori ceux qui sont le plus chargés, mais ce sont aussi ceux qui sont le plus inaccessibles, car en début de file. On aurait envie de réarranger les tenders pour enlever les tenders moins chargés qui le bloquent, mais notre contrainte est justement de ne pas le faire.

1.2 Résumé de l'étude

Dans la section 2, on s'intéresse à différents modèles dans lesquels le processus d'arrivée des clients est parfaitement régulier et déterministe. On y montre notamment que l'algorithme glouton est bien capable, dans le cadre déterministe, d'absorber le flux maximal de clients possible.

On introduit dans la section 3 le modèle probabiliste utilisé dans les sections suivantes. Dans ce modèle, les voitures arrivent de manière aléatoire, indépendamment des arrivées passées.

Dans la section 4, on utilise ce modèle pour étudier une station simplifiée, avec 4 tenders, 2 files et 3 niveaux de charge. On y montre que l'algorithme glouton est optimal lorsque les voitures n'arrivent en moyenne pas souvent, ou arrivent très souvent. Une simulation numérique montre que, pour une telle station, l'algorithme glouton est en réalité optimal quelle que soit la fréquence d'arrivée des voitures.

Dans la section 5, on développe des outils permettant de généraliser l'étude de la section précédente à tout nombre de tenders, de files ou de niveaux de charge. Ces outils permettent de montrer que pour les modèles avec deux files, l'algorithme glouton est optimal pour une fréquence d'arrivée des voitures très faible ou très élevée. Dans le cas intermédiaire où les voitures arrivent avec une fréquence "modérée", on montre que l'algorithme glouton n'est pas toujours optimal et peut être amélioré. Des simulations numériques permettent de donner une idée des valeurs de la fréquence moyenne d'arrivée des voitures pour laquelle l'algorithme glouton est optimal ou non. On conclut alors que l'algorithme optimal part de l'algorithme glouton, mais en dévie légèrement pour mettre en place un compromis entre longueur des files et niveau de charge. L'importance de cette déviation varie avec la fréquence d'arrivée des voitures.

Finalement, dans la section 6, on explore différents algorithmes implémentant en pratique ce compromis longueur/charge. Ces algorithmes reposent sur des indices : on associe à chaque file une valeur numérique qui dépend de sa longueur et de son niveau de charge, et on range les tenders dans la file où cette valeur est la plus faible. On trouve que les algorithmes ainsi obtenu permettent un gain en efficacité de l'ordre de quelques pourcents sur l'algorithme glouton, même dans le cas où la fréquence moyenne d'arrivée des clients n'est pas parfaitement connue.

2 Modèles déterministes

Avant de commencer des modèles probabilistes, plus avancées, inspectons ce que des modèles jouets peuvent nous apprendre sur notre problème.
On considère des modèles purement déterministes. On s'intéresse à une station de tenders. Cette station contient un stock de tenders, soit fini, soit infini (lorsqu'on veut ignorer cet aspect du problème). Les voitures clientes arrivent avec période T (c'est-à-dire une voiture cliente toutes les T unités de temps). Le flux de charge est τ : charger entièrement un tender complètement déchargé prend τ unité de temps. On considère f files où les tenders peuvent être chargés.

L'aspect du problème auquel on s'intéresse est celui du rangement, c'est-à-dire la manière dont on range les tenders lorsqu'ils arrivent en station, et dont on les prélève pour satisfaire la demande des voitures clientes. On considère donc des modèles avec rangement et des modèles sans rangement. Dans les modèles sans rangement, les tenders stationnent, virtuellement, "à côté" des files et n'y sont branchés qu'un par un, lorsqu'ils doivent être chargés. On peut donc toujours accéder à n'importe quel tender. À l'opposé, dans les modèles avec rangement, lorsque les tenders arrivent en station, ils doivent être branchés à une file. Les files fonctionnent sur une base "first in, last out" : lorsqu'un tender est branché à une file, il bloque l'accès aux tenders déjà branchés (et donc plus chargés). Les modèles sans rangement offrent un repère auquel on peut comparer les modèles avec rangement - on isole ainsi la part d'insatisfaction des clients qui est imputable au rangement à proprement parler.

Un autre aspect pris en compte dans l'analyse est celui du stock de tenders. En effet, les tenders peuvent s'épuiser en station. Lorsqu'on veut ignorer cet aspect, on considère que les voitures clientes amènent toutes avec elle un tender entièrement déchargé, qu'elles laissent derrière elles lorsqu'elles prélèvent un tender chargé. On a donc un stock constant de tenders en station. Si on souhaite prêter plus d'attention au stock, en revanche, on peut dissocier le flux entrant de tenders déchargés du flux sortant de tenders chargés. On intègre alors un paramètre λ indiquant qu'un tender entièrement déchargé entre dans la station toutes les λ unités de temps.

2.1 Modèle sans rangement, sans stock de tenders

Dans ce cas, on a f tenders chargés toutes les τ unités de temps (i.e. $\frac{f}{\tau}$ tenders chargés par unité de temps). La demande client correspond à $\frac{1}{T}$ tenders chargés par unité de temps. Tant que le stock de tenders initialement chargés est suffisant, la demande peut être satisfaite si $\frac{f}{\tau} \geq \frac{1}{T}$. À l'opposé, si $\frac{f}{\tau} < \frac{1}{T}$, le stock de tenders chargés finit par s'épuiser quelle que soit son importance initiale.

2.2 Modèle sans rangement, avec stock de tenders

Ici, un tender entre en station toutes les λ unités de temps, et un tender en sort toutes les T unités de temps. Ainsi, si $\lambda \leq T$, le stock de tenders en station ne s'épuise jamais. À l'opposé, si $\lambda > T$, le stock de tenders en station finit inexorablement par s'épuiser, et la demande ne pourra alors plus être satisfaite.
Puisque le problème du rangement ne se pose pas, tant qu'il y a des tenders en station, le problème ne diffère pas du modèle précédent. Dans le régime $\lambda \leq T$, un tender est chargé toutes les $\frac{\tau}{f}$ unités de temps en moyenne, et un tender chargé est prélevé toutes les T unités de temps en moyenne, donc la condition est la même que précédemment pour satisfaire les clients.

2.3 L'algorithme glouton pour le rangement

Dans les cas où un rangement est nécessaire pour les tenders, il nous faut décider d'un algorithme pour répartir les tenders déchargés qui arrivent entre les files, et pour décider d'où prélever les tenders chargés pour servir les clients.

L'algorithme qui nous intéresse le plus dans cette étude est celui que nous appellerons l'algorithme glouton. Il s'agit de l'algorithme naïf, consistant à ranger les tenders là où ils dérangent le moins, c'est à dire dans la file où le tender accessible est le moins chargé (pour ne pas bloquer l'accès aux tenders plus chargés des autres files).
De même les tenders sont prélevés dans la file contenant le moins de tenders parmi les files entièrement chargées.

Formellement, cet algorithme consiste à maximiser la durée pour laquelle on est capable de servir l'offre dans le pire des cas possible. (Il y a un parallèle à faire ici avec l'estimateur min-max en statistique). Pire veut dire : on suppose qu'à la période suivante, on a le plus grand nombre de clients qu'on est capable de servir si on a fait le choix le plus favorable - puis, parmi les choix compatibles, à la période suivante, on suppose qu'on a le plus grand nombre de clients possible, etc.
Dans la même logique, vider en priorité les files les moins remplies permet de les libérer plus vite. Ainsi, si toutes les files venaient à être remplies et chargées, nous en libérerions une plus vite et aurions donc moins de chance de devoir bloquer l'accès à des tenders déjà chargés en rangeant dans cette file un tender déchargé.

2.4 Modèle avec rangement, sans variation du stock de tenders

Si on note N le nombre, constant, de tenders en station, on a en moyenne $\left\lfloor \frac{N}{f} \right\rfloor$ tenders par file. On suppose donc qu'initialement, toutes les files sont entièrement chargées, avec $\left\lfloor \frac{N}{f} \right\rfloor$ ou $\left\lfloor \frac{N}{f} \right\rfloor + 1$ tenders.

En appliquant l'algorithme glouton, à la première voiture qui arrive, on va prélever un tender dans une des files les moins longues et placer le tender déchargé dans une autre des files les moins longues (parmi celles qui restent). La file où on a prélevé le tender est désormais moins longue que les autres - c'est donc dans celle-ci qu'on va continuer de prélever des tenders jusqu'à ce qu'elle s'épuise. De même, tant que $\tau > T$, la file où on a rangé le tender est moins chargée que les autres à l'arrivée de la voiture suivante, c'est donc dans celle-ci qu'on va continuer à ranger les tenders.
Une fois la première file vidée, elle devient prioritaire et c'est là qu'on va ranger les tenders. La file où on rangeait précédemment les tenders, qu'elle soit chargée ou non, est plus longue que les autres files, qui sont, elles, chargées. C'est donc dans la plus courte de ces dernières qu'on va aller prélever les tenders suivants, encore une fois jusqu'à épuisement. On remarque qu'il faut au moins trois files pour être capable de servir la demande.
À partir de là, on constate que l'algorithme adopte un comportement périodique. Après avoir vidé une file, il choisit la file la plus courte parmi celles qui restent, la vide tout en remplissant la file qu'il a vidé précédemment. Une fois cette file vidée, il passe à la suivante. Hormis la toute première file, qui a été remplie alors qu'elle n'était pas vide, toutes les files remplies ont la même longueur que la file qui a été vidée alors qu'elles étaient en train d'être remplies. Elles sont donc plus courtes que celles qui n'ont pas encore été tirées. Dès que l'une d'entre elles est chargée, l'algorithme ira reprendre des tenders dans cette file-là plutôt que dans celles pas encore exploitées.

Remarquons d'ores et déjà trois choses :

1. Ce comportement est intrinsèque à l'algorithme et ne dépend ni du caractère déterministe du modèle, ni du stock constant de tenders en station, ni de l'état initial. Tant qu'il n'y a pas d'intervalle de temps de durée supérieure ou égale à τ où aucune voiture cliente n'arrive, l'algorithme va vider les files une par une et remplir la file vide.
2. L'algorithme commence en fait par vider complètement une file pour créer un espace de stockage des tenders déchargés. On se retrouve donc avec une file deux fois plus longue que les autres. En fait, mieux vaut commencer dans un état initial avec une file vide et $f-1$ files contenant $\left\lfloor \frac{N}{f-1} \right\rfloor$

ou $\left\lfloor \frac{N}{f-1} \right\rfloor + 1$ tenders. On considère cet état initial dans la suite, et on suppose que toutes les files non-vides ont $n = \frac{N}{f-1} \in \mathbb{N}$ tenders.

3. Dans le modèle déterministe, l'algorithme utilise le nombre minimal de files pour assurer la tournante, c'est à dire qu'il n'entame de nouvelles files que tant que la première file qu'il a remplie n'est pas encore chargée. Dans le cadre déterministe, il n'y a donc pas d'intérêt à avoir un nombre de files supérieur. Dans le cadre où l'arrivée des clients n'est pas complètement connue et régulière, on ne sait pas combien de voitures arriveront avant que la première file soit chargée. Ainsi, plus le nombre de files est grand, plus on est capable d'absorber les périodes de demande plus intenses auxquelles le hasard nous confronte.

On peut concevoir le processus de chargement des tenders de deux manières. Soit tous les tenders branchés à une file se chargent en même temps, avec une durée τ pour charger chacun d'entre eux, indépendamment du nombre de tenders branchés ; soit chaque file ne charge qu'au plus k tender à la fois (les plus profonds dans la file parmi ceux qui ne sont pas complètement chargés). Nous nous concentrerons dans cette section sur le cas où chaque file ne peut charger qu'un tender à la fois, et référons au lecteur à la section 2.6 pour une présentation des résultats dans les autres cas.

Déterminons maintenant sous quelles conditions l'algorithme glouton parvient à satisfaire la demande. Depuis le moment où le premier tender déchargé est placé dans une file vide, il faut une durée de chargement τ par tender pour recharger complètement la file, et aucun tender ne peut être prélevé avant car les tenders sont chargés par ordre d'arrivée (donc les tenders chargés seront bloqués par les autres). Comme on va mettre n tenders dans la file en tout, il faudra une durée $n\tau$ pour tous les charger. Or, au moment de l'arrivée du premier tender déchargé, c'est la N-ième prochaine voiture qui devra s'alimenter dans la file où il a été placé, et elle arrive dans une durée NT. On conclut que l'algorithme glouton permet de satisfaire la demande tant que :

$$n\tau \leq NT \qquad \text{i.e.} \qquad \frac{\tau}{f-1} \leq T$$

Ainsi, par rapport au cas sans rangement, le rangement fait perdre le bénéfice d'exactement une file pour le chargement. $\frac{1}{\tau}$ représente la puissance de charge d'une file, ainsi, la condition précédente impose que la station ait une puissance de charge supérieure à la fréquence d'arrivée des clients ; par rapport au cas sans rangement, on perd la contribution d'une file à la puissance de charge de la station. On remarque que le stock de tenders ne joue aucun rôle ici sur la capacité à soutenir la demande sur le long-terme.
En fait, aucun algorithme ne peut satisfaire la demande sur le long-terme dans le cas où $\tau > T(f-1)$. En effet, on perd des tenders chargés avec la même fréquence que l'arrivée de voiture cliente, c'est-à-dire $\frac{1}{T}$. Chaque file charge les tenders déchargés avec flux de charge $\frac{1}{\tau}$. Il y a certes f files, mais en fait, si l'on considère la dernière file où un tender a été prélevé, comme aucun tender n'a pu y être ajouté depuis, elle est soit entièrement chargée soit vide. Dans tous les cas, la file n'a donc aucun tender à charger. À tout instant, au plus $f-1$ files contribuent à la charge des tenders et le flux de charge de la station est donc au plus $\frac{f-1}{\tau}$. La demande ne pourra donc pas être satisfaite sur le long terme si $\frac{f-1}{\tau} < T$.

On a donc, en un certain sens, optimalité de l'algorithme glouton. Cet algorithme est capable de soutenir la plus haute intensité de demande possible, bien que d'autres algorithmes puissent également présenter cette propriété.

La taille des files oscillent entre 0 et n de manière stable, un atout de cet algorithme est qu'on évite d'avoir des files trop longues ou de longueurs trop inégales.

Remarque pour l'implémentation : Ici, pour simplifier, on a supposé que lorsque le dernier tender d'une file était prélevé on ne pouvait pas y ranger simultanément le tender déchargé qui vient d'arriver.

En fait, en pratique, c'est possible, et même souhaitable puisque cela évite d'avoir une file vide, qui ne contribue donc pas à la charge des tenders. Cela ne change pas le comportement de l'algorithme, et intuitivement ne devrait pas remettre en question son optimalité une fois ce changement mis en place. Cependant, le gain obtenu est minimal tant que les files ont des tailles non-négligeables : la file en cours d'exploitation passera le plus clair de son temps complètement chargée, et non vide.

2.5 Modèle avec rangement et variation du stock de tenders

On finit en considérant le cas pour lequel, dans le modèle précédent, l'arrivée des tenders est décorrélée de l'arrivée des voitures clientes. Comme vu précédemment, si $\lambda > T$, le stock de tender en station s'épuise linéairement avec pente $\frac{1}{T} - \frac{1}{\lambda}$. L'algorithme glouton devrait cependant pouvoir assurer l'approvisionnement des clients jusqu'à épuisement des tender, car la longueur des files successives est à peu près décroissante (la seule source de problème est le fait que les divisions ne sont pas entières).

À l'opposé, si $\lambda < T$, on surremplit les files. Le temps pour nous de vider une file de n tenders, on aura rempli la file vide avec $\left\lceil \frac{nT}{\lambda} \right\rceil > n$ tenders au plus. Comme les files se succèdent de plus en plus longues, on va passer une durée d'au moins $n(f-1)T$ avant de s'intéresser à la file. La condition pour que l'algorithme glouton puisse soutenir la demande sur le long terme devient donc :

$$\left\lceil \frac{nT}{\lambda} \right\rceil \tau \leq n(f-1)T$$

qui s'approxime par

$$\frac{\tau}{f-1} \leq \lambda$$

Le stock de tender grandit donc avec intensité $\frac{1}{\lambda} - \frac{1}{T}$ à condition que la puissance de charge soit suffisante pour charger des tenders supplémentaires, au-delà de simplement maintenir le stock de tenders constant.

Remarque pour l'implémentation : Si le problème se posait, il est possible que commencer dans un état initial où toutes les files ne sont pas de la même longueur mais s'enchaînent comme une suite géométrique de raison $\left(\frac{T}{\lambda}\right)^{\frac{1}{f-1}}$ parvienne à régler ce problème. On laisse cette considération au lecteur.

2.6 Généralisation à d'autres modèles

Jusque là, on s'est concentré sur le cas où chaque file ne pouvait charger qu'un tender à la fois. Cependant, en pratique, une file peut charger jusqu'à $k \geq 1$ tenders en même temps. Si la taille des files est en général proche de ou plus petit que k, le modèle dans lequel une file charge tous les tenders qui y sont branchés en même temps peut être pertinent.
Les calculs faits précédemment s'adaptent bien à ces nouveaux modèles. Ils sont laissés au lecteur et on présente simplement leurs résultats.

Dans le cas où les files chargent tous leurs tenders en même temps, la condition pour que l'algorithme glouton soit capable de soutenir la demande est :

$$\tau \leq T\left(1 + N\,\frac{f-2}{f-1}\right)$$

Le stock a alors une importance. En effet, une fois remplie, une file a besoin d'un temps constant, τ, pour être entièrement chargée, mais plus il y a de tenders en station, plus on peut attendre longtemps avant d'avoir besoin des tenders de cette file.
On n'est cependant pas capable de montrer dans ce cas que l'algorithme glouton permet la valeur minimale de T.

Dans le cas où les files peuvent charger jusqu'à k tenders en même temps, et en supposant pour simplifier que le nombre de tenders en station N est un multiple de $k(f-1)$, la condition pour que l'algorithme glouton soit capable de soutenir la demande est donnée par :

$$\frac{\tau}{k(f-1)} \leq T\left(1 - \frac{k-1}{N}\right)$$

On retrouve une formule proche de celle trouvée dans le cas où $k = 1$, mais τ est divisée par un facteur k : la puissance de charge des files est multipliée par k. Le stock apparaît dans la formule, mais tant que le nombre de tenders en station est grand par rapport au nombre de tenders qu'une file est capable de charger simultanément, la condition se réduit approximativement à :

$$\frac{\tau}{k(f-1)} \leq T$$

Un faible stock de tenders en station est pénalisant, car on n'est alors pas en capacité de permettre aux files d'avoir k tenders déchargés à charger en même temps.
Notre argument précédent ne permet pas d'établir que la première contrainte doit être vérifiée pour tout algorithme capable de soutenir la demande, mais permet d'établir la seconde contrainte.

2.7 Conclusions générales

Ainsi, ces modèles jouets nous ont d'ores et déjà appris un certain nombre de choses sur notre problème. Tout d'abord, pour que la demande puisse être soutenue sur le long terme, la puissance de charge totale de la station doit être au-delà d'un certain seuil. En pratique, la puissance de charge d'une file donnée peut difficilement être augmentée, ainsi, si l'on souhaite pouvoir accueillir plus de clients, il faut augmenter le nombre de files en station.

Le fait de devoir ranger les tenders dans une file, et donc de bloquer l'accès à certains tenders, pénalise la capacité d'une station à servir les clients. Cette pénalité, par rapport au cas sans rangement (tous les tenders sont constamment accessibles), est représentée par la perte d'une file. Intuitivement, à tout moment, une file doit être entièrement chargée afin de pouvoir servir le prochain client, et donc ne charge pas les tenders. La capacité de charge de la station est limitée comme si elle avait une file de moins.
On remarque au passage que l'on pourrait faire l'économie d'une file en station en retirant tous les tenders d'une file dès qu'elle est entièrement chargée et en les stockant sur le côté. Dans ce cas, toutes les files sont constamment utilisées au maximum de leur capacité.

La condition pour savoir si une station comportant f files est capable de soutenir une demande avec une voiture qui arrive après chaque intervalle de temps T, et τ dénote le temps pour qu'une file charge complètement un tender, est :

$$\begin{aligned} \tau &\leq Tf && \text{dans le cas sans rangement} \\ \tau &\leq T(f-1) && \text{dans le cas avec rangement} \end{aligned}$$

On suppose ici que chaque file n'est capable de charger qu'un tender à la fois, celui le plus profond dans la file. Si une file peut charger k tenders à la fois, une bonne approximation de la nouvelle contrainte consiste à diviser τ par k dans la contrainte précédente.

L'algorithme glouton offre l'exemple d'un algorithme capable de soutenir une telle demande. Cet algorithme est optimal dans le sens où il est capable de soutenir la plus haute fréquence possible pour l'arrivée des clients. En revanche, ce n'est pas forcément le seul algorithme à avoir cette propriété, et d'autres algorithmes sont peut-être plus robustes à l'irrégularité dans l'arrivée des clients - c'est-à-dire à un processus d'arrivée des clients aléatoire, avec le même intervalle de temps moyen entre deux voitures. Ce cas sera

étudié plus avant dans les sections suivantes.

Finalement, on remarque que le stock de tenders en station ne joue pas sur la soutenabilité de la demande sur le long terme : peu importe qu'il y ait beaucoup ou peu de tenders en station. Cela s'explique par le caractère déterministe des modèles utilisés. En pratique, ce que cela veut dire, c'est qu'augmenter le stock de tenders dans une station n'augmente pas la capacité à soutenir sur le long-terme une demande plus intense. En revanche, cela peut aider à absorber le choc d'une augmentation temporaire de la demande. Heuristiquement, si, en moyenne, un tender arrive en station tous les intervalles de temps λ, alors la station perd (si $\lambda > T$) ou gagne (si $\lambda < T$) un tender après chaque intervalle de temps $\frac{\lambda T}{|\lambda - T|}$. La demande peut être servie tant qu'il reste des tenders en station. Ainsi, on peut tolérer un pic d'activité où $\lambda > T$ tant que la durée de ce pic n'est pas suffisante pour épuiser le stock de tenders. On peut ensuite espérer $\lambda < T$ pour reconstituer les stocks, mais alors dans le cas où le problème de rangement se pose, il faut vérifier que le critère ci-dessus marche toujours lorsqu'on remplace T par λ, pour s'assurer que la station a une capacité de charge suffisante pour absorber ces nouveaux tenders.

On se place dans toute la suite dans le cas où les files chargent tous les tenders qui y sont branchés simultanément. Cela est une bonne approximation de la réalité tant que les files restent de petite taille par rapport au nombre de tenders qu'elles peuvent charger simultanément.

3 Introduction au modèle probabiliste

On n'a pour l'instant considéré qu'un modèle déterministe, dans lequel la demande est parfaitement connue et prévisible à l'avance. En pratique, ce n'est pas le cas, et le fait qu'on ne sache pas en avance quand et en quel nombre les prochains clients arriveront rend plus difficile de servir la demande : on doit ranger les tenders de sorte à avoir toujours des tenders chargés prêts à servir un éventuel client qui arriverait. Pour capturer cette difficulté, on va désormais construire un modèle avec un processus d'arrivée des clients aléatoire.

3.1 Description du modèle

On considère une station comportant f files, pour un total de n tenders en circulation. Chaque tender comporte $m+1$ états de charge, numérotés de 0 à 1. À 0 un tender est complètement déchargé, et à 1 il est complètement chargé.

On utilise un modèle à temps discret.

Les voitures clientes arrivent de sorte que l'écart temporel entre la i-ième voiture et la $i+1$-ième soit de loi géométrique de paramètre θ : c'est la seule manière de modéliser une arrivée des voitures sans mémoire (sachant qu'aucune voiture n'est arrivée au temps t, la probabilité qu'une voiture soit arrivée avant le temps $t+t'$ est la même que la probabilité (non conditionnelle) qu'une voiture arrive avant le temps t'). Lorsqu'elles arrivent, elles doivent prendre un tender complètement chargé qu'elles ramènent complètement déchargé au début de l'étape suivante.

Fatalement, un jour plus de n voitures arriveront en même temps et alors la demande ne pourra pas être satisfaite. On note $T_0 = T_{\text{ERREUR}}$ (les deux notations seront utilisées) le premier temps où la demande en tender est insatisfaite et on s'intéresse à son espérance. On cherche à trouver, pour un processus de rangement et de sélection des tenders donné, l'espérance $\mathbb{E}_i(T_0)$ en partant d'un état i à préciser, qu'on souhaite maximal.

Chaque étape temporelle se déroule comme suit :

1. Tous les tenders dans la station gagnent un niveau de charge, pour un niveau maximum de 1.
2. Les tenders partis à l'étape précédente sont rangés dans la station.
3. Les nouvelles voitures clientes arrivent et des tenders entièrement chargés leur sont attribués.
4. Si la demande a pu être satisfaite, on passe au temps suivant.

Notre modèle prend alors la forme d'une chaîne de Markov : les états sont les dispositions possibles de la station et la probabilité de transition d'un arrangement κ_1 vers un arrangement κ_2 est donnée par la probabilité qu'un nombre adéquat de voitures arrive au temps 0, transformant l'arrangement κ_1 vers l'arrangement κ_2 en utilisant un algorithme donné. Ce modèle est bien Markovien.

En cas de doute sur des notion de chaînes Markoviennes, on pourra se référer aux notes de cours de Jean-François Le Gall [Gal06].

3.2 Liaison entre les paramètres du modèles et les paramètres réels

Les deux équations suivantes sont fondamentales : si τ_v est le temps moyen entre l'arrivée de 2 voitures dans le modèle, on a

$$\tau_v = \frac{1-\theta}{\theta}$$

et le temps de charge du modèle τ_c vérifie

$$\tau_c = m.$$

Ces deux équations sont fondamentales car elles permettent de calibrer le modèle aux paramètres du monde réel. Dans notre modèle, on utilise un temps discret, avec les étapes temporelles successives numérotées par des entiers. L'écart temporel entre deux étapes successives est en quelque sorte la "nouvelle unité de temps" qu'on définit pour notre modèle. En pratique, on connaît, au moins approximativement, le temps moyen réel entre deux clients, noté $\tau_{v,real}$, ainsi que le temps de charge réel d'un tender, $\tau_{c,real}$. On connaît donc le ratio des deux, qui ne dépend pas de l'unité utilisée puisque c'est un nombre sans unité :

$$\frac{\tau_{v,real}}{\tau_{c,real}} = \frac{\tau_v}{\tau_c}.$$

Ainsi, on commence par choisir m pour décider de la résolution temporelle du modèle, le plus petit intervalle de temps qu'on est capable de distinguer. Une unité de temps dans notre modèle correspond alors à une durée réelle $\frac{\tau_{c,real}}{m}$, et on détermine la bonne valeur de θ pour notre modèle grâce à la formule précédente :

$$\frac{1-\theta}{\theta} = \tau_c \frac{\tau_{v,real}}{\tau_{c,real}}$$

qui donne

$$\theta = \frac{1}{1 + \tau_c \frac{\tau_{v,real}}{\tau_{c,real}}} \tag{1}$$

On a bien $\theta \in]0,1[$.

La phase $\theta \to 1$ correspond à une très grande fréquence d'arrivée des voitures et $\theta \to 0$ correspond à une période creuse où les voitures se font rares.

4 Étude du modèle à 4 tenders, 2 files et 3 niveaux de charge

On s'intéresse dans cette section au modèle à 4 tenders, 2 files et 3 niveaux de charge ($f = 2$, $n = 4$, $m = 2$), car c'est le modèle le plus simple pour lequel une transition n'est pas triviale à décider.

Précisons quelques notations : un état du système sera de la forme $\begin{matrix} * & i \\ * & j \end{matrix}$ ou $\begin{matrix} & & i & \\ * & * & & j \end{matrix}$ avec $i, j \in \{0, 0.5, 1\}$, avec les $*$ pouvant être des états de charge quelconques qui n'ont pas d'importance : les tenders les plus au fond sont ceux arrivés le plus tôt, ils sont donc les plus chargés et pour pouvoir être atteints, il faut que ceux devant aient été enlevés, donc ils seront nécessairement chargés à 1.

Dans ce modèle, toutes les transitions sont triviales à décider, sauf une : lorsqu'une seule voiture arrive, et qu'on se trouve dans l'état $\begin{matrix} * & 1 \\ * & 0 \end{matrix}$, alors on a 2 choix possibles : soit on replace le tender de la manière suivante : $\begin{matrix} & & 1 & \\ * & * & & 0 \end{matrix}$ (ce qui crée un modèle de chaîne de Markov, qu'on référera par premier modèle), soit on le replace de la manière suivante : $\begin{matrix} * & 0 \\ * & 0{,}5 \end{matrix}$ (qu'on référera par second modèle).

La transition n'est pas évidente à déterminer car dans la première transition on a encore un tender disponible directement mais au temps suivant on a encore seulement qu'un tender disponible, alors que dans la seconde transition, on a pas de tender disponible directement mais au temps suivant, on a 2 tenders disponibles au lieu d'un.

Malheureusement, un calcul direct semble voué à l'échec dû à la complexité de la chaîne et son absence de symétries. Bien sûr, en théorie il est possible de prendre la matrice de transition et de l'inverser, mais le calcul serait extrêmement lourd (il faudrait inverser une matrice de l'ordre de 15×15, et cela pour un modèle très simple !) et très peu éclairant. On va donc étudier successivement deux cas : le cas θ proche de 0 (les voitures arrivent peu souvent) et le cas θ proche de 1 (les voitures arrivent très fréquemment).

4.1 Phase basse fréquence

L'idée des cette phase est d'analyser trajectoire par trajectoire ce qui se passe : lorsque θ tend vers 0, les trajectoires avec beaucoup de voitures sont trop "coûteuses" (improbables) et sont donc négligeables, et la chaîne est alors simplifiée.

Formalisons cela. On note E l'ensemble des états de la chaîne, état ERREUR exclus.

On utilisera la formule suivante pour l'espérance :

$$\mathbb{E}_{x_0}(T_{\text{ERREUR}}) = \sum_{x_1,\dots,x_{n-1} \in E} nQ(x_0, x_1) \dots Q(x_{n-2}, x_{n-1})Q(x_{n-1}, \text{ERREUR})$$

où $x_0 \in E$ est fixé et Q est la matrice de transition de la chaîne.

4.1.1 Cas du premier algorithme

On commence par traiter le cas de la chaîne 1, mais les calculs sont identiques pour la chaîne 2 et il suffira juste de changer les "inputs" liés à la matrice de transition de la matrice.

On voit le processus comme une marche aléatoire Markovienne sur les états possibles de la station. L'idée est alors de dire que la marche reste souvent en $\alpha := \begin{matrix} * & 1 \\ * & 1 \end{matrix}$ ou en $\beta := \begin{matrix} * & * & 1 \\ 1 & & \end{matrix}$.

Commençons par introduire une notation : on écrit $x(\theta) \propto y(\theta)$ si $\frac{x(\theta)}{y(\theta)}$ converge vers un $a > 0$ lorsque θ tend vers 0.

On va montrer que

$$a(\theta) \leq \mathbb{E}_{x_0,\theta,1}(T_{\text{ERREUR}}) \leq b(\theta) \quad \text{avec } a(\theta), b(\theta) \propto \frac{1}{\theta^4}.$$

Remarque. *À partir de ça, on peut en fait montrer plus précisément que* $\mathbb{E}_{x_0,\theta,1}(T_{\text{ERREUR}}) \propto \frac{1}{\theta^4}$ *puisque le vecteur* $v = (\mathbb{E}_{x,\theta,1}(T_{\text{ERREUR}}))_{x\in E}$ *est solution d'un système linéaire* $v = A(\theta)e$ *où* $e = \begin{pmatrix} 1 \\ \vdots \\ 1 \\ 0 \end{pmatrix}$ *et* $A(\theta)$ *est une matrice polynomiale en* θ*, donc* $v = \frac{{}^t Com A(\theta)}{\det A(\theta)} e$ *est un vecteur fraction rationnel en* θ*. On ne détaillera pas plus cela ici car cela ne présente pas d'intérêt particulier pour l'étude.*

Notons $T_1 > 0, \ldots, T_n$ les temps successifs où la marche est en $\alpha = \begin{matrix} * & 1 \\ * & 1 \end{matrix}$ ou en $\beta = \begin{matrix} * & * & 1 \\ & 1 & \end{matrix}$, avec T_n le dernier instant où la marche va en un de ces deux états avant d'aller en erreur.

On note

$$\varphi_{ij}(\theta) := \mathbb{P}_i(T_1 < \infty, X_{T_1} = j),$$

pour $i, j = \alpha$ ou β, et

$$\Phi_i(\theta) := \mathbb{P}_i(T_1 = \infty).$$

On écrit alors pour $i \in \{\alpha, \beta\}$,

$$\mathbb{E}_i(T_{\text{ERREUR}}) = \mathbb{E}_i(T_1 \mathbb{1}_{T_1<\infty}) + \varphi_{i\alpha}\mathbb{E}_\alpha(T_{\text{ERREUR}}) + \varphi_{i\beta}\mathbb{E}_\beta(T_{\text{ERREUR}}) + \mathbb{E}_i(T_{\text{ERREUR}} \mathbb{1}_{T_1=\infty})$$

en conditionnant avec le temps d'arrêt T_1, en appliquant la propriété de Markov forte.

On trouve alors le système

$$\begin{pmatrix} \mathbb{E}_\alpha(T_0) \\ \mathbb{E}_\beta(T_0) \\ \mathbb{E}_0(T_0) \end{pmatrix} = \begin{pmatrix} \mathbb{E}_\alpha(T_1 \mathbb{1}_{T_1<\infty}) + \mathbb{E}_\alpha(T_{\text{ERREUR}} \mathbb{1}_{T_1=\infty}) \\ \mathbb{E}_\beta(T_1 \mathbb{1}_{T_1<\infty}) + \mathbb{E}_\beta(T_{\text{ERREUR}} \mathbb{1}_{T_1=\infty}) \\ 0 \end{pmatrix} + \begin{pmatrix} \varphi_{\alpha\alpha}(\theta) & \varphi_{\alpha\beta}(\theta) & 0 \\ \varphi_{\beta\alpha}(\theta) & \varphi_{\beta\beta}(\theta) & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \mathbb{E}_\alpha(T_0) \\ \mathbb{E}_\beta(T_0) \\ \mathbb{E}_0(T_0) \end{pmatrix}$$

donc

$$\begin{aligned} \begin{pmatrix} \mathbb{E}_\alpha(T_0) \\ \mathbb{E}_\beta(T_0) \\ \mathbb{E}_0(T_0) \end{pmatrix} &= \begin{pmatrix} 1-\varphi_{\alpha\alpha}(\theta) & -\varphi_{\alpha\beta}(\theta) & 0 \\ -\varphi_{\beta\alpha}(\theta) & 1-\varphi_{\beta\beta}(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix}^{-1} \begin{pmatrix} \mathbb{E}_\alpha(T_1 \mathbb{1}_{T_1<\infty}) + \mathbb{E}_\alpha(T_{\text{ERREUR}} \mathbb{1}_{T_1=\infty}) \\ \mathbb{E}_\beta(T_1 \mathbb{1}_{T_1<\infty}) + \mathbb{E}_\beta(T_{\text{ERREUR}} \mathbb{1}_{T_1=\infty}) \\ 0 \end{pmatrix} \\ &= \frac{\begin{pmatrix} 1-\varphi_{\beta\beta}(\theta) & \varphi_{\alpha\beta}(\theta) & 0 \\ \varphi_{\beta\alpha}(\theta) & 1-\varphi_{\alpha\alpha}(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \mathbb{E}_\alpha(T_1 \mathbb{1}_{T_1<\infty}) + \mathbb{E}_\alpha(T_{\text{ERREUR}} \mathbb{1}_{T_1=\infty}) \\ \mathbb{E}_\beta(T_1 \mathbb{1}_{T_1<\infty}) + \mathbb{E}_\beta(T_{\text{ERREUR}} \mathbb{1}_{T_1=\infty}) \\ 0 \end{pmatrix}}{(1-\varphi_{\alpha\alpha}(\theta))(1-\varphi_{\beta\beta}(\theta)) - \varphi_{\alpha\beta}(\theta)\varphi_{\beta\alpha}(\theta)} \end{aligned}$$

Il faut cependant montrer que les espérances sont bien finies. On écrit

$$\mathbb{E}_i(T_1 \mathbb{1}_{T_1<\infty}) = \sum_{n\geq 0} \mathbb{P}_i(T_1 > n)$$

$$
\begin{aligned}
&\leq \sum_{n\geq 0} \mathbb{P}(N_1 \leq 3, \ldots, N_{\lfloor \frac{n}{3} \rfloor} \leq 3) \\
&= \sum_{n\geq 0} 3\mathbb{P}(N_1 \leq 3)^n \\
&= 3\sum_{n\geq 0} (1-(1-\theta)^4)^n \\
&= \frac{3}{(1-\theta)^4} \\
&\leq 4 \quad \text{pour } \theta \text{ assez petit}
\end{aligned}
$$

car pour la seconde ligne, si $T_1 > n$ alors on a au moins une voiture qui vient tous les 3 unités de temps (car sinon on retombe avec tous les tenders pleins dans α ou β). On déduit de même que

$$\mathbb{E}_i(T_0 \mathbb{1}_{T_1=\infty}) \leq 1 \text{ pour } \theta \text{ assez petit.}$$

Mais on peut écrire $1-\varphi_{\beta\beta}(\theta) = \varphi_{\beta\alpha}(\theta) + \Phi_\beta(\theta)$ et $1-\varphi_{\alpha\alpha}(\theta) = \varphi_{\alpha\beta}(\theta) + \Phi_\alpha(\theta)$, et pour θ assez petit,

$$1 \leq \mathbb{E}_\alpha(T_1 \mathbb{1}_{T_1<\infty}) + \mathbb{E}_\alpha(T_{\text{ERREUR}} \mathbb{1}_{T_1=\infty}) \leq 5 \tag{2}$$

et

$$1 \leq \mathbb{E}_\beta(T_1 \mathbb{1}_{T_1<\infty}) + \mathbb{E}_\beta(T_{\text{ERREUR}} \mathbb{1}_{T_1=\infty}) \leq 5 \tag{3}$$

donc

$$\frac{\begin{pmatrix} \varphi_{\alpha\beta}(\theta)+\varphi_{\beta\alpha}(\theta)+\Phi_\beta(\theta) \\ \varphi_{\alpha\beta}(\theta)+\varphi_{\beta\alpha}(\theta)+\Phi_\alpha(\theta) \\ 0 \end{pmatrix}}{(1-\varphi_{\alpha\alpha}(\theta))(1-\varphi_{\beta\beta}(\theta)) - \varphi_{\alpha\beta}(\theta)\varphi_{\beta\alpha}(\theta)} \leq \begin{pmatrix} \mathbb{E}_\alpha(T_0) \\ \mathbb{E}_\beta(T_0) \\ \mathbb{E}_0(T_0) \end{pmatrix} \leq \frac{\begin{pmatrix} 5(\varphi_{\alpha\beta}(\theta)+\varphi_{\beta\alpha}(\theta)+\beta_0\theta^5+O(\theta^6)) \\ 5(\varphi_{\alpha\beta}(\theta)+\varphi_{\beta\alpha}(\theta)+\alpha_0\theta^4+O(\theta^5)) \\ 0 \end{pmatrix}}{(1-\varphi_{\alpha\alpha}(\theta))(1-\varphi_{\beta\beta}(\theta)) - \varphi_{\alpha\beta}(\theta)\varphi_{\beta\alpha}(\theta)} \tag{4}$$

dans le sens où cela est vrai pour toutes les coordonnées. Explicitons maintenant les spécificités de l'algorithme :

$$\begin{aligned}
\varphi_{\alpha\beta}(\theta) &\propto \theta^2, \\
\varphi_{\beta\alpha}(\theta) &\propto \theta^2, \\
\Phi_\alpha(\theta) &\propto \theta^4, \\
\Phi_\beta(\theta) &\propto \theta^5.
\end{aligned}$$

Nous laissons au lecteur le soin de vérifier que le choix de l'algorithme 1, c'est à dire de l'algorithme glouton, résulte en effet en de telles estimations.

De ces équations, on déduit

$$\begin{aligned}
(1-\varphi_{\alpha\alpha}(\theta))(1-\varphi_{\beta\beta}(\theta)) - \varphi_{\alpha\beta}(\theta)\varphi_{\beta\alpha}(\theta) &= (\Phi_\alpha(\theta) + \varphi_{\alpha\beta}(\theta)\Phi_\beta(\theta) + \varphi_{\beta\alpha}(\theta)) - \varphi_{\alpha\beta}(\theta)\varphi_{\beta\alpha}(\theta) \\
&\propto \theta^6
\end{aligned}$$

et

$$\varphi_{\alpha\beta}(\theta) + \varphi_{\beta\alpha}(\theta) \propto \theta^2,$$

donc on peut en déduire finalement

$$a(\theta) \leq \mathbb{E}_i(T_{\text{ERREUR}}) \leq b(\theta) \quad \text{où } a(\theta), b(\theta) \propto \frac{1}{\theta^4}, \text{ et } i = \alpha, \beta$$

Cependant, il n'est pas difficile de voir que quitte à ajouter un premier retour en α ou β, cela ne va pas changer l'ordre de grandeur, donc l'estimation précédente est en fait vraie pour tout $i \in E$.

4.1.2 Cas du second algorithme

On utilise les mêmes notations que dans la section précédente, mais cette fois-ci les spécificités de l'algorithme donnent

$$\varphi_{\alpha\beta}(\theta) \propto \theta^3,$$
$$\varphi_{\beta\alpha}(\theta) \propto \theta^2,$$
$$\Phi_{\alpha}(\theta) \propto \theta^3,$$
$$\Phi_{\beta}(\theta) \propto \theta^5.$$

Les inégalités 2, 3 et 4 restent vraies (elles ont été obtenues avec des principes qui ne dépendent pas des spécificités de l'algorithme), ce qui permet d'écrire

$$\begin{aligned}(1-\varphi_{\alpha\alpha}(\theta))(1-\varphi_{\beta\beta}(\theta)) - \varphi_{\alpha\beta}(\theta)\varphi_{\beta\alpha}(\theta) &= (\Phi_\alpha(\theta)+\varphi_{\alpha\beta}(\theta))(\Phi_\beta(\theta)+\varphi_{\beta\alpha}(\theta)) - \varphi_{\alpha\beta}(\theta)\varphi_{\beta\alpha}(\theta) \\ &\propto \theta^5\end{aligned}$$

et

$$\varphi_{\alpha\beta}(\theta) + \varphi_{\beta\alpha}(\theta) \propto \theta^2,$$

ce qui implique finalement

$$a(\theta) \leq \mathbb{E}_i(T_{\text{ERREUR}}) \leq b(\theta) \quad \text{où } a(\theta), b(\theta) \propto \frac{1}{\theta^3}, \text{ et } i = \alpha, \beta$$

Cependant, il est similairement facile de voir que quitte à ajouter un premier retour en 0, cela ne va pas changer l'ordre de grandeur, donc l'estimation précédente est en fait vraie pour tout $i \in E$.

4.1.3 Conclusion

On en déduit donc finalement qu'a fortiori, $\mathbb{E}_{x_0,\theta,1}(T_{\text{ERREUR}}) > \mathbb{E}_{x_0,\theta,2}(T_{\text{ERREUR}})$ pour θ assez proche de 0, mais c'est en fait beaucoup plus fort que cela car leurs ordres de grandeur diffèrent d'un ordre $\frac{1}{\theta}$.

4.2 Phase haute fréquence

Pour tout modèle de la forme décrite précédemment,

On écrit $\mathbb{E}_{\alpha,\theta}(T_0) = \sum_{n\geq 0} \mathbb{P}_\alpha(T_0 > n)$ où α est un état de la chaîne, qu'on omettra dans les quelques prochaines formules générales.

Mais on a $\mathbb{P}(T_0 = n+1 \mid T_0 > n) \geq \theta^5$ puisque partant de n'importe quel état, si 5 voitures arrivent, on est envoyé en erreur. Donc

$$\mathbb{P}(T_0 > n) - \mathbb{P}(T_0 > n+1) = \mathbb{P}(T_0 = n+1) \geq \theta^5 \mathbb{P}(T_0 > n)$$

D'où

$$\mathbb{P}(T_0 > n+1) \leq (1-\theta^5)\mathbb{P}(T_0 > n)$$

Donc

$$\left| \sum_{n\geq n_0} \mathbb{P}(T_0 > n) \right| \leq \sum_{n\geq n_0} (1-\theta^5)^n \mathbb{P}(T_0 > 0) \leq \frac{(1-\theta^5)^{n_0}}{1-(1-\theta^5)} = \frac{(1-\theta^5)^{n_0}}{\theta^5} \tag{5}$$

Donc pour θ suffisamment loin de 0 et de 1, pour comparer deux modèles, il suffit de comparer les premiers termes du développement de $\sum_{n\geq 0} \mathbb{P}(T_0 > n)$.

4.2.1 Borne explicite

Regardons donc pour nos deux modèles ce que donne cette méthode : on regarde pour $\alpha = \begin{matrix} * & 1 \\ * & 0 \end{matrix}$ l'expansion de la somme à 4 termes pour commencer pour le premier modèle (noté 1 en indice dans l'espérance) et le second (noté 2 en indice de l'espérance)
On a pour $i \in \{1, 2\}$,

$$\mathbb{E}_{i,\theta}(T_0) = \mathbb{P}_i(T_0 > 0) + \mathbb{P}_i(T_0 > 1) + \sum_{n \geq 2} \mathbb{P}_i(T_0 > n)$$

$$= (1-\theta^3) + 1 - (\theta^3 + \mathbb{P}(T_0 = 1 \mid N_1 > 0)\mathbb{P}(N_1 > 0) + \mathbb{P}_i(T_0 = 1 \mid N_1 = 0, N_2 > 0)\mathbb{P}(N_1 = 0, N_2 > 0)$$

$$+\mathbb{P}_i(T_0 = 1 \mid N_1 = N_2 = 0, N_3 > 0)\mathbb{P}(N_1 = N_2 = 0, N_3 > 0) + \sum_{n \geq 2} \mathbb{P}_i(T_0 > n)$$

Mais ici, le seul terme indépendant de i est $\mathbb{P}_i(T_0 = 1 \mid N_1 = 0, N_2 > 0)$ et

$$\mathbb{P}_1(T_0 = 1 \mid N_1 = 0, N_2 > 0)\mathbb{P}(N_1 = 0, N_2 > 0) = \mathbb{P}(N_1 = 0, N_2 = 1, N_3 = 0) = \theta^3(1-\theta)$$

et

$$\mathbb{P}_2(T_0 = 1 \mid N_1 = 0, N_2 > 0) = \mathbb{P}(N_1 = 0, N_2 = 1) = \theta^2(1-\theta)$$

On conclut alors que lorsque $\theta^2(1-\theta)^2 > 2\frac{(1-\theta^5)^2}{\theta^5}$, alors $\mathbb{E}_{2,\alpha,\theta}(T_0) < \mathbb{E}_{1,\alpha,\theta}(T_0)$. Cette inégalité n'est malheureusement jamais vérifiée.
Poussons alors un petit peu plus le développement : voyons les différences pour

$$\mathbb{P}_i(T_0 > 2) = 1 - \mathbb{P}_i(T_0 = 0) - \mathbb{P}_i(T_0 = 1) - \mathbb{P}_i(T_0 = 2)$$

On a

$$\mathbb{P}_1(T_0 = 2 \mid N_1 = 0, N_2 > 0)\mathbb{P}(N_1 = 0, N_2 > 0) = \mathbb{P}(N_1 = 0, N_2 = N_3 = 1) + \mathbb{P}(N_1 = 0, N_2 = 2, N_3 = 0)$$

$$= \theta^3(1-\theta)^2 + \theta^3(1-\theta)^2 = 2\theta^3(1-\theta)^2$$

et

$$\mathbb{P}_2(T_0 = 2 \mid N_1 = 0, N_2 > 0)\mathbb{P}(N_1 = 0, N_2 > 0) = \mathbb{P}(N_1 = 0, N_2 = 2, N_3 = 0, N_4 = 0) = \theta^4(1-\theta)^2$$

Donc si

$$\theta^2(1-\theta)(2 + \theta^2(1-\theta) - 2\theta(2-\theta)) = 2\theta^2(1-\theta) + \theta^4(1-\theta)^2 - 2\theta^3(1-\theta) - 2\theta^3(1-\theta)^2 > 2\frac{(1-\theta^5)^3}{\theta^5}$$

Alors

$$\mathbb{E}_{2,\alpha,\theta}(T_0) < \mathbb{E}_{1,\alpha,\theta}(T_0)$$

En résolvant la dernière inégalité graphiquement, on déduit que pour $\theta \geq 0.9961$, on a bien $\mathbb{E}_{2,\alpha,\theta}(T_0) < \mathbb{E}_{1,\alpha,\theta}(T_0)$.

 **Remarque.** On aimerait cependant montrer cette inégalité pour tout θ : cette inégalité semble être vraie par simulation informatique pour tout $\theta \in [0, 1]$. Si ceci est bien vrai, en poussant de plus en plus le développement, on obtient l'inégalité voulue pour des valeurs arbitrairement petites de θ. Cela demande cependant rapidement une capacité importante de calcul.

4.2.2 Équivalent

On développe à l'ordre 2 $\mathbb{E}_{i,\alpha,\theta}(T_0)$ en $1-\theta$ avec l'expression $\mathbb{E}_{\alpha,\theta}(T_0) = \sum_{n\geq 0} \mathbb{P}_\alpha(T_0 > n)$:

$$\mathbb{E}_{1,\alpha,\theta}(T_0) - \mathbb{E}_{2,\alpha,\theta}(T_0) = \theta(1-\theta)\theta - \theta(1-\theta)\theta^2 + \sum_{n\geq 2} \mathbb{P}_1(T_0 > n) - \sum_{n\geq 2} \mathbb{P}_2(T_0 > n)$$

Mais

$$\left| \sum_{n\geq 2} \mathbb{P}_1(T_0 > n) - \sum_{n\geq 2} \mathbb{P}_2(T_0 > n) \right| \leq \frac{(1-\theta^5)^2}{\theta^5}$$

Mais $\frac{(1-\theta^5)^2}{\theta^5} = O\left((1-\theta)^2\right)$, donc on retrouve bien la difficulté qu'on a eu précédemment : on est obligé de pousser le développement pour obtenir quelque chose.

On écrit alors

$$\begin{aligned}\mathbb{E}_{1,\alpha,\theta}(T_0) - \mathbb{E}_{2,\alpha,\theta}(T_0) = 2(\theta(1-\theta)\theta - \theta(1-\theta)\theta^2) + \theta(1-\theta)\left((1-\theta)\theta^3 - 2\theta^2(1-\theta)\right) \\ + \sum_{n\geq 3} \mathbb{P}_1(T_0 > n) - \sum_{n\geq 3} \mathbb{P}_2(T_0 > n) \quad (6)\end{aligned}$$

avec

$$\left| \sum_{n\geq 3} \mathbb{P}_1(T_0 > n) - \sum_{n\geq 3} \mathbb{P}_2(T_0 > n) \right| \leq \frac{(1-\theta^5)^3}{\theta^5} = O\left((1-\theta)^3\right)$$

donc

$$\begin{aligned}\mathbb{E}_{1,\alpha,\theta}(T_0) - \mathbb{E}_{2,\alpha,\theta}(T_0) &= 2(1-\theta)^2\theta^2 + (1-\theta)^2(\theta^4 - 2\theta^3) + O\left((1-\theta)^3\right) \\ &= (1-\theta)^2 + O\left((1-\theta)^3\right)\end{aligned}$$

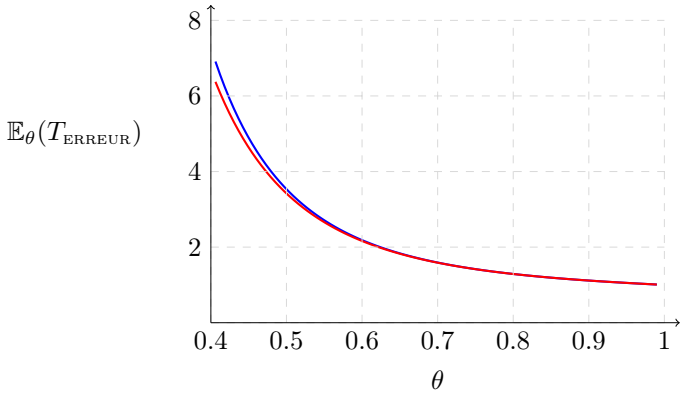

FIGURE 1 – Graphe de la fonction $\theta \mapsto \mathbb{E}_\theta(T_{\text{ERREUR}})$. La courbe bleue représente cette fonction pour l'algorithme glouton (qui correspond au premier modèle, l'algorithme glouton envoie par définition chaque tender à la file ayant la charge la plus petite de tender devant, et parmi celles-ci à la longueur minimale), et la courbe rouge représente cette fonction pour le second modèle. L'inégalité semble être vraie pour tout $\theta \in [0.4, 1]$, et elle est vraie sur toutes les valeurs de $\theta \in [0.4, 1]$ utilisées pour tracer cette figure.

4.3 Conclusion

Les deux études basses et hautes fréquences semblent converger sur le fait que le modèle 1 est meilleur que le modèle 2. En fait, lorsqu'on fait des simulations informatiques, la domination $\mathbb{E}_{1,\alpha,\theta}(T_0) > \mathbb{E}_{2,\alpha,\theta}(T_0)$ semble être vraie pour tout $\theta \in [0,1]$. Les figures 1 et 2 montrent cette domination.

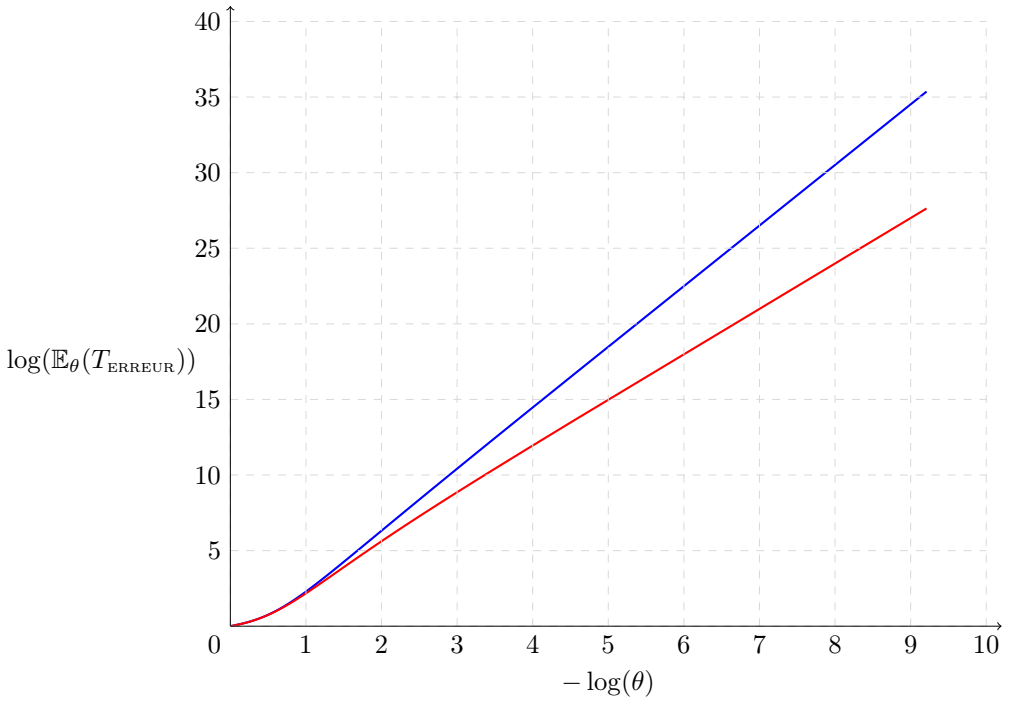

FIGURE 2 – Graphe en échelle logarithmique de la fonction $\theta \mapsto \mathbb{E}_\theta(T_{\text{ERREUR}})$. La légende est la même que pour la figure précédente. On retrouve alors la domination de l'algorithme glouton (premier modèle) sur le second modèle, pour les faibles valeurs de θ, et on voit aussi le comportement en puissance de $\frac{1}{\theta}$ lorsque $\theta \to 0$.

5 Généralisation à des modèles plus complets

La section précédente montre quelque chose d'intéressant et de non évident à première vue : à l'arrivée d'une seule voiture, la transition $\begin{matrix} * & 1 \\ * & 0 \end{matrix} \to \begin{matrix} 1 & \\ * & * & 0 \end{matrix}$ est fondalement meilleure que la transition $\begin{matrix} * & 1 \\ * & 0 \end{matrix} \to \begin{matrix} * & 0 \\ * & 0{,}5 \end{matrix}$. Ce n'est pas évident à première vue car dans la première transition on a encore un tender disponible directement mais au temps suivant on a encore seulement qu'un tender disponible, alors que dans la seconde transition, on a pas de tender disponible directement mais au temps suivant, on a 2 tenders disponibles au lieu d'un. Mais dans le modèle d'arrivée géométrique des voitures, donc arrivée sans mémoire, il se trouve que la première transition domine toujours la seconde.

Cependant ce modèle n'est pas très avancé car il n'a que 3 niveaux de charge des tenders et ne contient que 4 tenders. Le but est alors ici d'étudier des modèles similaires de répartition des tenders, mais qui contiennent plus de niveaux de charge (toujours un nombre fini cependant), et plus de tenders.

L'étude du cas précédent a été cependant pensée pour être généralisable, et c'est donc cela qu'on va traiter dans cette section : on va généraliser les résultats et méthodes de la partie précédente à des modèles plus généraux de la forme précédente, en suivant le même parcours que dans la section 4

5.1 Phase basse fréquence

La phase basse fréquence était la phase la plus difficile à étudier dans la première section, et elle va rester la plus difficile à généraliser. On partira cependant des mêmes idées que dans la partie basse fréquence de la première section.

Fixons le cadre général dans lequel on va étudier le modèle : on note Stab les états stables de la chaîne (ceux qui ne sont composés que de 1), $T_1, \dots, T_n$ les temps successifs où la marche X_i vérifie $X_i \in \text{Stab}$ et on note de même, pour $i, j \in \text{Stab}$,

$$\varphi_{ij}(\theta) := \mathbb{P}_i(T_1 < \infty, X_{T_1} = j)$$

et

$$\Phi_i(\theta) := \mathbb{P}_i(T_1 = \infty)$$

On omettra parfois la dépendance en θ en écrivant $\varphi_{ij} = \varphi_{ij}(\theta)$, et de même pour Φ.

On montre de même que dans la section 4.1, équations 2 et 3, que

$$\forall i \in \text{Stab}, \quad 1 \leq \mathbb{E}_i(T_1 \mathbb{1}_{T_1<\infty}) + \mathbb{E}_i(T_{\text{ERREUR}} \mathbb{1}_{T_1=\infty}) \leq A \tag{7}$$

autour de $\theta = 0$, avec $A > 0$ fixé, et on en déduit alors pour $i \in \text{Stab}$,

$$\mathbb{E}_i(T_{\text{ERREUR}}) = \mathbb{E}_i(T_1 \mathbb{1}_{T_1<\infty}) + \sum_{j \in \text{Stab}} \varphi_{ij} \mathbb{E}_j(T_{\text{ERREUR}}) + \mathbb{E}_i(T_{\text{ERREUR}} \mathbb{1}_{T_1=\infty})$$

en conditionnant avec le temps d'arrêt T_1, en appliquant la propriété de Markov forte.
On a alors le même système linéaire que dans la première section :

$$\begin{pmatrix} \mathbb{E}_{i_1}(T_0) \\ \vdots \\ \mathbb{E}_{i_n}(T_0) \\ \mathbb{E}_0(T_0) \end{pmatrix} = \begin{pmatrix} \mathbb{E}_{i_1}(T_1 \mathbb{1}_{T_1<\infty}) + \mathbb{E}_{i_1}(T_{\text{ERREUR}} \mathbb{1}_{T_1=\infty}) \\ \vdots \\ \mathbb{E}_{i_n}(T_1 \mathbb{1}_{T_1<\infty}) + \mathbb{E}_{i_n}(T_{\text{ERREUR}} \mathbb{1}_{T_1=\infty}) \\ 0 \end{pmatrix} + \begin{pmatrix} \varphi_{i_1 i_1} & \dots & \varphi_{i_1 i_n} & 1 - \varphi_{i_1 i_1} - \dots - \varphi_{i_1 i_n} \\ \vdots & \ddots & \vdots & \vdots \\ \varphi_{i_n i_1} & \dots & \varphi_{i_n i_n} & 1 - \varphi_{i_n i_1} - \dots - \varphi_{i_n i_n} \\ 0 & \dots & 0 & 0 \end{pmatrix} \begin{pmatrix} \mathbb{E}_{i_1}(T_0) \\ \vdots \\ \mathbb{E}_{i_n}(T_0) \\ \mathbb{E}_0(T_0) \end{pmatrix}$$

donc

$$\begin{pmatrix} 1-\varphi_{i_1i_1} & \dots & -\varphi_{i_1i_n} & 1-\varphi_{i_1i_1}-\dots-\varphi_{i_1i_n} \\ \vdots & \ddots & \vdots & \vdots \\ -\varphi_{i_ni_1} & \dots & 1-\varphi_{i_ni_n} & 1-\varphi_{i_ni_1}-\dots-\varphi_{i_ni_n} \\ 0 & \dots & 0 & 1 \end{pmatrix} \begin{pmatrix} \mathbb{E}_{i_1}(T_0) \\ \vdots \\ \mathbb{E}_{i_n}(T_0) \\ \mathbb{E}_0(T_0) \end{pmatrix} = \begin{pmatrix} \mathbb{E}_{i_1}(T_1\mathbb{1}_{T_1<\infty}) + \mathbb{E}_{i_1}(T_{\text{ERREUR}}\mathbb{1}_{T_1=\infty}) \\ \vdots \\ \mathbb{E}_{i_n}(T_1\mathbb{1}_{T_1<\infty}) + \mathbb{E}_{i_n}(T_{\text{ERREUR}}\mathbb{1}_{T_1=\infty}) \\ 0 \end{pmatrix}$$

ainsi

$$\begin{pmatrix} 1-\varphi_{i_1i_1} & -\varphi_{i_1i_2} & \dots & -\varphi_{i_1i_n} \\ -\varphi_{i_2i_1} & 1-\varphi_{i_2i_2} & \dots & -\varphi_{i_2i_n} \\ \vdots & \vdots & \ddots & \vdots \\ -\varphi_{i_ni_1} & -\varphi_{i_ni_2} & \dots & 1-\varphi_{i_ni_n} \end{pmatrix} \begin{pmatrix} \mathbb{E}_{i_1}(T_0) \\ \vdots \\ \mathbb{E}_{i_n}(T_0) \end{pmatrix} = \begin{pmatrix} \mathbb{E}_{i_1}(T_1\mathbb{1}_{T_1<\infty}) + \mathbb{E}_{i_1}(T_{\text{ERREUR}}\mathbb{1}_{T_1=\infty}) \\ \vdots \\ \mathbb{E}_{i_n}(T_1\mathbb{1}_{T_1<\infty}) + \mathbb{E}_{i_n}(T_{\text{ERREUR}}\mathbb{1}_{T_1=\infty}) \end{pmatrix}$$

et on va utiliser les formules de Cramer afin de déterminer l'expression des $\mathbb{E}_{i_j}(T_0)$.

On pose $\Phi_i(\theta) := 1-\varphi_{ii_1}-\dots-\varphi_{ii_n} = \mathbb{P}_i^{\mathbb{CM}_0}(T_1=\infty)$, et on écrit

$$\begin{pmatrix} \Phi_{i_1}+\sum\limits_{j\neq i_1}\varphi_{i_1j} & -\varphi_{i_1i_2} & \dots & -\varphi_{i_1i_n} \\ -\varphi_{i_2i_1} & \Phi_{i_2}+\sum\limits_{j\neq i_2}\varphi_{i_2j} & \dots & -\varphi_{i_2i_n} \\ \vdots & \vdots & \ddots & \vdots \\ -\varphi_{i_ni_1} & -\varphi_{i_ni_2} & \dots & \Phi_{i_n}+\sum\limits_{j\neq i_n}\varphi_{i_nj} \end{pmatrix} \times \begin{pmatrix} \mathbb{E}_{i_1}(T_0) \\ \vdots \\ \mathbb{E}_{i_n}(T_0) \end{pmatrix} = \begin{pmatrix} \mathbb{E}_{i_1}(T_1\mathbb{1}_{T_1<\infty}) + \mathbb{E}_{i_1}(T_{\text{ERREUR}}\mathbb{1}_{T_1=\infty}) \\ \vdots \\ \mathbb{E}_{i_n}(T_1\mathbb{1}_{T_1<\infty}) + \mathbb{E}_{i_n}(T_{\text{ERREUR}}\mathbb{1}_{T_1=\infty}) \end{pmatrix}$$

La formule de Cramer s'écrit

$$\mathbb{E}_{i_1}(T_0) = \frac{\begin{vmatrix} \mathbb{E}_{i_1}(T_1\mathbb{1}_{T_1<\infty}) + \mathbb{E}_{i_1}(T_{\text{ERREUR}}\mathbb{1}_{T_1=\infty}) & -\varphi_{i_1i_2} & \dots & -\varphi_{i_1i_n} \\ \mathbb{E}_{i_2}(T_1\mathbb{1}_{T_1<\infty}) + \mathbb{E}_{i_2}(T_{\text{ERREUR}}\mathbb{1}_{T_1=\infty}) & \Phi_{i_2}+\sum\limits_{j\neq i_2}\varphi_{i_2j} & \dots & -\varphi_{i_2i_n} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbb{E}_{i_n}(T_1\mathbb{1}_{T_1<\infty}) + \mathbb{E}_{i_n}(T_{\text{ERREUR}}\mathbb{1}_{T_1=\infty}) & -\varphi_{i_ni_2} & \dots & \Phi_{i_n}+\sum\limits_{j\neq i_n}\varphi_{i_nj} \end{vmatrix}}{\begin{vmatrix} \Phi_{i_1}+\sum\limits_{j\neq i_1}\varphi_{i_1j} & -\varphi_{i_1i_2} & \dots & -\varphi_{i_1i_n} \\ -\varphi_{i_2i_1} & \Phi_{i_2}+\sum\limits_{j\neq i_2}\varphi_{i_2j} & \dots & -\varphi_{i_2i_n} \\ \vdots & \vdots & \ddots & \vdots \\ -\varphi_{i_ni_1} & -\varphi_{i_ni_2} & \dots & \Phi_{i_n}+\sum\limits_{j\neq i_n}\varphi_{i_nj} \end{vmatrix}} \tag{8}$$

et de même symétriquement pour les autres états de Stab.

Notre travail est alors maintenant de comprendre ces deux déterminants.

Commençons par le déterminant au dénominateur. On écrit

$$\begin{vmatrix} \Phi_{i_1}+\sum\limits_{j\neq i_1}\varphi_{i_1j} & -\varphi_{i_1i_2} & \dots & -\varphi_{i_1i_n} \\ -\varphi_{i_2i_1} & \Phi_{i_2}+\sum\limits_{j\neq i_2}\varphi_{i_2j} & \dots & -\varphi_{i_2i_n} \\ \vdots & \vdots & \ddots & \vdots \\ -\varphi_{i_ni_1} & -\varphi_{i_ni_2} & \dots & \Phi_{i_n}+\sum\limits_{j\neq i_n}\varphi_{i_nj} \end{vmatrix} = \sum_{\sigma\in\mathfrak{S}_n}\varepsilon(\sigma)\prod_{1\leq i\leq n}a_{i\sigma(i)}$$

où les (a_{ij}) sont les éléments de la matrice. En développant tous les termes, on trouve qu'il n'y a que des termes de la forme $\pm \prod_{1\leq i\leq n} b_{i,f(i)}$ avec $f : \{1,\dots,n\} \to \{1,\dots,n\}$ et $b_{i,j} = \varphi_{ij}$ si $j \geq 1$ et $b_{i,i} = \Phi_i$.

Une façon de voir ces termes est de les associer à un graphe orienté de sommets $\{0,\dots,n\}$ qui a pour arêtes les indices associés aux facteurs, comme dans la figure 3.

FIGURE 3 – La figure de gauche représente le produit $\varphi_{12}\Phi_2\varphi_{34}\Phi_4$, et celle de droite le produit $\varphi_{12}\varphi_{23}\varphi_{34}\varphi_{41}$.

Remarquons que la forme des termes de la somme impose que tous les sommets du graphe ont exactement une arête sortante.

Notre but est alors de compter le nombre de fois que le terme $\prod_{1\leq i\leq n} b_{i,f(i)}$ apparait dans la somme, pour une f fixée.

Supposons qu'il existe $m \geq 1$ cycles dans le graphe associé (qui sont de sommets disjoints car tous les sommets du graphe ont exactement une arête sortante). Alors il apparaît dans le terme $\sigma = \mathrm{id}$ avec un signe $+$, et pour tout cycle σ_0 du graphe, il apparaît dans le terme $\sigma = \sigma_0$ avec un signe $-$ car $\varepsilon(\sigma_0) = (-1)^{l+1}$ où l est le cardinal du support du cycle, et on trouve un autre produit de l (-1) car il y a le produit $(-\varphi_{i_1i_2})(-\varphi_{i_2i_3})\dots(-\varphi_{i_li_1})$ et les autre facteurs sont positifs. En faisant cela avec les permutations du type $\sigma = \sigma_1\dots\sigma_k$ où les σ_i sont des cycles du graphe, on trouve qu'il y a exactement

$$\sum_{0\leq i\leq m} \binom{m}{i}(-1)^i = (1-1)^m = 0$$

fois ce terme au final dans la somme.

Donc tous les termes contenant un cycle sont éliminés dans la somme et il ne reste que des termes dont le graphe est de la forme du graphe de gauche dans la figure 3, c'est à dire des graphes avec plusieurs chemins disjoints menant uniquement à l'état ERREUR, puisque tous les sommets du graphe ont exactement une arête sortante. Ils n'apparaissent donc que dans le terme $\sigma = \mathrm{id}$ et ils ont donc un signe $+$ et on a alors une façon pratique de calculer l'équivalent du déterminant : il suffit de prendre le terme de plus bas degré en θ.

On continue donc avec le déterminant au numérateur.

La figure 4 montre un terme typique dans l'expression du déterminant du numérateur : l'état 1 est imbriqué dans un chemin qui ne boucle pas et qui ne va pas en erreur, tandis que les états qui ne sont pas dans ce chemin sont dans le même cas que précédemment : soit ils sont imbriqués dans un cycle, soit ils sont imbriqués dans un chemin menant en ERREUR. Du reste, on montre de même que précédemment que lorsqu'il existe un cycle, ne contenant pas l'état 1 donc, les contributions de ce terme dans la somme de l'expression du déterminant du nominateur s'annulent et ce terme n'apparait pas dans l'expression finale

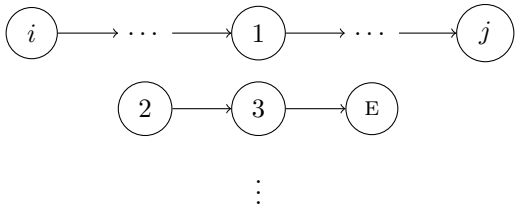

FIGURE 4 – Terme typique apparaissant dans l'expression du déterminant du numérateur, où ici on a posé $i_1 = 1$.

développée.

On déduit finalement que l'expression finale, développée, ne contient que des termes dont les graphes associés sont de la forme de celui montré dans la figure 4, avec seulement un signe +, car soit ils apparaissent dans le terme $\sigma = \mathrm{id}$, soit dans un σ qui est un cycle contenant 1 mais dans ce cas là, il y a pour les signes $\varepsilon(\sigma) = (-1)^{l+1}$ et $(-1)^{l-1}$ car le produit $(-\varphi_{i_1 i_2})(-\varphi_{i_2 i_3})\dots(-\varphi_{i_l i_1})$ est un produit de seulement $l-1$ $(-\varphi_{ij})$ et d'un 1 (qui correspondait à $(-\varphi_{1j})$ dans le cas du dénominateur). On a donc aussi une manière pratique de calculer l'équivalent du déterminant du numérateur.

5.2 Phase haute fréquence

La phase haute fréquence est elle beaucoup plus facile à généraliser. Supposons être dans un modèle à n tenders. Alors en raisonnant de même que dans la section 4.2, pour tout état i du modèle, $\mathbb{P}_i(T_0 = m+1 \mid T_0 > m) \geq \theta^{n+1}$ et on en déduit de même

$$\left| \sum_{m \geq n_0} \mathbb{P}_i(T_0 > m) \right| \leq \frac{(1-\theta^{n+1})^{n_0}}{\theta^{n+1}}. \tag{9}$$

Et pour comparer deux modèles, il suffit de même de prendre les premiers termes du développement de $\mathbb{E}_i(T_0)$ pour obtenir un résultat soit sous forme explicite, comme dans la section 4.2.1, mais avec plus de calcul, ou sous forme d'équivalent, comme dans la section 4.2.2.

5.3 Résolution du modèle à n tenders, 2 files et $m+1$ niveaux de charge

On se munit ici d'un modèle avec n tenders, m niveaux de charge, et 2 files.

On va montrer que lorsqu'une seule voiture arrive, la transition

$$\begin{array}{cccccc} * & \dots & * & \frac{m-2}{m} & \to & * \quad \dots \quad * \quad 0 \\ * & 1 & & & & 1 \end{array}$$

qu'on associe à une première chaîne de Markov, est meilleure que la transition

$$\begin{array}{cccccc} * & \dots & * & \frac{m-2}{m} & \to & * \quad \dots \quad * \quad \frac{m-1}{m} \\ * & 1 & & & & * \quad 0 \end{array} \tag{10}$$

qu'on associe à une seconde chaîne de Markov, ce qui suffira pour décrire toutes les transitions (toutes les autres transitions a priori non triviales seront en quelque sorte des affaiblissements de cette transition). On s'apprette donc à étudier 2 modèles exhibant ces transitions, mais les transitions

$$\begin{array}{ccccccc} (i) & * & \dots & \dots & * & \frac{k}{m} & \to \quad ? \\ (j) & * & \dots & * & 1 & & \end{array}$$

ne sont aussi pas évidentes à décider si $j < i$, $k < m$ et une seule voiture arrive (les (i) et (j) représentent le nombre de tenders dans la file). Pour la section 5.3.1, on supposera pour le premier modèle que toutes ces transition seront

$$\begin{array}{cccccc} (i) & * & \dots & \dots & * & \frac{k}{m} \\ (j) & * & \dots & * & 1 & \end{array} \rightarrow \begin{array}{cccccc} (i+1) & * & \dots & \dots & * & 0 \\ (j-1) & * & \dots & * & 1 & \end{array} \tag{11}$$

et pour le second modèle on supposera aussi que toutes les transitions sont de la forme 11, sauf dans le cas $k = m-1$, $j = 2$ et $i = n-2$ où la transition prend bien évidemment la forme de 10. En fait, cela va être assez clair que transformer des transitions de la forme de 11 dans des transitions de la forme de 10 ne ferait que rendre le résultat encore plus fort (cela distancerait les 2 modèles d'un ordre encore plus grand que $\frac{1}{\theta}$).

5.3.1 Phase basse fréquence

On va utiliser les résultats de la section 5.1 afin d'analyser le comportement des 2 modèles lorsque $\theta \to 0$.

Commençons la preuve dans le cas de 2 files.

On note, pour $i \leq \frac{n}{2}$, stab_i l'état $\begin{array}{ccccc} * & \dots & * & * & 1 \\ * & \dots & * & 1 & \end{array}$ état qui contient i tenders dans la file du bas et $n-i$ dans la file du haut, et on note, pour $l = 1, 2$, Q_l la matrice de transition de la chaîne de Markov associée aux états de Stab de la chaîne l ($Q_l(i,j) := \varphi_{ij}$ et $Q_l(i,0) := \Phi_i$).

On a pour $l = 1, 2$,

$$Q_l(\text{stab}_i, \text{stab}_{i+1}) \propto \theta^2,$$

$$\forall i \neq j \leq \frac{n}{2}, \quad Q_l(\text{stab}_i, \text{stab}_j) \propto \theta^k \quad \text{avec } k \geq 2,$$

$$\forall i \leq \frac{n}{2}, \quad Q_1(\text{stab}_i, 0) \propto \theta^{1+n-i+1} = \theta^{n-i+2},$$

$$\forall i \leq \frac{n}{2}, \quad Q_2(\text{stab}_i, 0) \propto \theta^{1+n-i-2+1+1} = \theta^{n-i+1},$$

les 2 premières estimations sont assez claires, et pour les 2 dernières, dans le cas 1, on doit d'abord bloquer la file la moins grande en prenant un premier tender tout seul, puis vider entièrement l'autre file en la déversant dans la file initialement moins grande puis reprendre un tender pour finir en ERREUR, et dans le cas 1, on doit d'abord bloquer la file la moins grande en prenant un premier tender tout seul, puis vider l'autre file jusque laisser seulement 2 tenders chargés, attendre que l'autre file ait son dernier tender en charge $\frac{m-2}{m}$, puis reprendre un tender, attend une unité de temps, puis reprendre un tender pour finir en ERREUR.

Donc dans l'équation 8, pour la chaîne 1, pour le déterminant au dénominateur, le chemin optimal est de ne prendre qu'un chemin qui va en erreur, et c'est en fait assez clair que c'est le chemin

$$\text{stab}_0 \to \text{stab}_1 \to \text{stab}_2 \to \cdots \to \text{stab}_{\lfloor \frac{n}{2} \rfloor} \to 0,$$

qui parcourt tous les états de Stab, et qui induit un terme en

$$\theta^{2(\lfloor \frac{n}{2} \rfloor + 1) + n - \lfloor \frac{n}{2} \rfloor + 2},$$

et pour le déterminant au nominateur, c'est le chemin

$$\text{stab}_0 \to \text{stab}_1 \to \text{stab}_2 \to \cdots \to \text{stab}_{\lfloor \frac{n}{2} \rfloor}$$

qui est optimal (et cela ne dépend pas de l'état de Stab duquel on part), et qui induit un terme en

$$\theta^{2(\lfloor \frac{n}{2} \rfloor + 1)},$$

donc pour tous les états γ de la chaîne 1,

$$\mathbb{E}_{1,\gamma,\theta}(T_{\text{ERREUR}}) \propto \frac{1}{\theta^{n-\lfloor \frac{n}{2} \rfloor + 2}}$$

Pour la chaîne 2, les chemins optimaux sont les mêmes chemins, mais le changement d'estimation pour aller en erreur induit que le dénominateur est en

$$\theta^{2(\lfloor \frac{n}{2} \rfloor + 1) + n - \lfloor \frac{n}{2} \rfloor + 1},$$

et le nominateur aussi en

$$\theta^{2(\lfloor \frac{n}{2} \rfloor + 1)},$$

donc pour tous les états γ de la chaîne 1,

$$\mathbb{E}_{2,\gamma,\theta}(T_{\text{ERREUR}}) \propto \frac{1}{\theta^{n-\lfloor \frac{n}{2} \rfloor + 1}}$$

Donc au final, on a bien lorsque $\theta \to 0$,

$$\mathbb{E}_{2,\gamma,\theta}(T_{\text{ERREUR}}) < \mathbb{E}_{1,\gamma,\theta}(T_{\text{ERREUR}}).$$

On a donc montré que l'algorithme glouton est meilleur que l'autre algorithme pour θ assez petit. On peut en déduire que l'algorithme glouton est meilleurs que tous les autres algorithmes pour θ assez petit. Nous avons donc montré la domination de l'algorithme glouton pour tous les modèles à 2 files pour θ assez petit.

Nous nous sommes placés dans le cas à 2 files pour simplifier la preuve, mais le résultat reste en fait vrai pour un nombre de files quelconque : sous des hypothèses raisonnables, l'algorithme glouton est meilleur que tous les autres algorithmes pour θ assez petit, à n, m, et f fixés.

Remarque. Cela peut sembler assez contre-intuitif, voire même complètement faux lorsque l'on prend la limite $m \to \infty$ ou $n \to \infty$ ou même $m, n \to \infty$, mais les « domaines de validité » en θ des équivalents trouvés dans cette partie dépendent de m et n et vont typiquement être extrêmement réduits autour de 0 lorsque $m \to \infty$ ou $n \to \infty$. C'est ce qui sera étudié dans la section 5.3.3. On peut tout de même mentionner que ce résultat n'est vraiment pas si absurde car avec θ très petit, le fait de ne pas avoir de tender disponible est une grande vulnérabilité car il suffit d'une seule voiture pour aller en erreur, donc une transition d'un ordre θ. Avec un tender disponible cependant, il est beaucoup plus improbable d'aller en erreur car il faut donc 2 voitures, donc un coût beaucoup plus important de θ^2, même si cela « saborde » une file très grande de tenders quasi chargés.

5.3.2 Phase haute fréquence

On applique le raisonnement de la partie 5.2 pour avoir un équivalent. En faisant exactement les mêmes calculs que dans l'équation 6, on déduit

$$\begin{aligned}\mathbb{E}_{1,\alpha,\theta}(T_0) - \mathbb{E}_{2,\alpha,\theta}(T_0) = 2\theta(1-\theta)\theta - 2\theta(1-\theta)\theta^2 - 2\theta(1-\theta)\theta^2(1-\theta) + \theta(1-\theta)\theta^{n-1}(1-\theta) \\ + \sum_{n\geq 3} \mathbb{P}_1(T_0 > n) - \sum_{n\geq 3} \mathbb{P}_2(T_0 > n)\end{aligned}$$

donc puisque

$$\left| \sum_{n\geq 3} \mathbb{P}_1(T_0 > n) - \sum_{n\geq 3} \mathbb{P}_2(T_0 > n) \right| \leq \frac{(1-\theta^{n+1})^3}{\theta^{n+1}} = O((1-\theta)^3)$$

on déduit

$$\begin{aligned} \mathbb{E}_{1,\alpha,\theta}(T_0) - \mathbb{E}_{2,\alpha,\theta}(T_0) &= 2(1-\theta)^2\theta^2 + (1-\theta)^2(\theta^n - 2\theta^3) + O((1-\theta)^3) \\ &= (1-\theta)^2 + O((1-\theta)^3) \end{aligned}$$

ce qui conclut $\mathbb{E}_{1,\alpha,\theta}(T_0) > \mathbb{E}_{2,\alpha,\theta}(T_0)$ pour θ assez proche de 1.

Remarque. Ce résultat ne doit pas avoir beaucoup d'importance d'un point de vue industriel : il est vrai seulement lorsque θ est très grand, mais dans ces plages de θ, c'est surtout la station qui est largement sous-dimensionnée par rapport à la demande. Il parait alors absurde de vouloir servir les clients instantanément quitte à bloquer des tenders presque chargés. Dans de tels cas, on cherchera plutôt à dimensionner adéquatement la station, et avant que ce dimensionnement soit mis en place, on implémentera plutôt l'algorithme glouton qui permettra de servir les clients avec en moyenne un temps d'attente minimal (mais grand tout de même à cause du sous-dimensionnement de la station).

5.3.3 Phase moyenne fréquence

On va étudier ici une forme de réponse au questionnement posé à la fin de la section 5.3.1. On va essayer de montrer un résultat du type : si on se donne deux bornes $0 < \theta_1 < \theta_2 < 1$, alors on peut trouver $m, n \in \mathbb{N}$ tels que

$$\mathbb{E}_{2,\gamma_2,\theta}(T_{\text{ERREUR}}) > \mathbb{E}_{1,\gamma_1,\theta}(T_{\text{ERREUR}}), \quad \text{pour tout } \theta \in [\theta_1, \theta_2]$$

avec

$$\gamma_1 = \begin{matrix} * & \dots & * & 0 \\ 1 & & & \end{matrix},$$

et

$$\gamma_2 = \begin{matrix} * & \dots & * & \frac{m-1}{m} \\ * & 0 & & \end{matrix},$$

puis on en déduira que pour tous γ état du système,

$$\mathbb{E}_{2,\gamma,\theta}(T_{\text{ERREUR}}) > \mathbb{E}_{1,\gamma,\theta}(T_{\text{ERREUR}}), \quad \text{pour tout } \theta \in [\theta_1, \theta_2].$$

On écrit alors

$$\mathbb{E}_{2,\gamma_2,\theta}(T_{\text{ERREUR}}) \geq (1-\theta)\mathbb{E}(N_1 + \cdots + N_n)$$

avec (N_i) iid de loi $N_1 \sim \text{Geom}(\theta)$, puisque s'il n'y a pas de voiture à l'instant 0, alors les n premières voitures ne peuvent créer d'erreur.

Donc

$$\mathbb{E}_{2,\gamma_2,\theta}(T_{\text{ERREUR}}) \geq n\frac{(1-\theta)}{\theta} \geq n\frac{1-\theta_2}{\theta_2}.$$

Et on écrit pour la première chaîne, en utilisant la borne 9 utilisée en haute fréquence lorsque la grande file est rechargée, on déduit

$$\mathbb{E}_{1,\gamma_1,\theta}(T_{\text{ERREUR}}) \leq ((1-\theta)^{m+1} + m(1-\theta)^{m+1}\theta)\left(m + 1 + \frac{1}{\theta^{n+1}}\right) + \sum_{0\leq k\leq m} k \times k\theta^2(1-\theta)^k$$

puisque $\mathbb{P}(N_1 + N_2 \geq m+1) = (1-\theta)^{m+1} + m(1-\theta)^{m+1}\theta$.

Donc

$$\begin{aligned}
\mathbb{E}_{1,\gamma_1,\theta}(T_{\text{ERREUR}}) &\leq ((1-\theta)^{m+1} + m(1-\theta)^{m+1}\theta)\left(m+1+\frac{1}{\theta^{n+1}}\right) + \sum_{1\leq k\leq m}(k+1)(k+2)\theta^2(1-\theta)^k \\
&\leq (1-\theta)^{m+1}(1+m\theta)\left(m+1+\frac{1}{\theta^{n+1}}\right) + \theta^2\frac{2}{(1-(1-\theta))^3} \\
&\leq (1-\theta)^{m+1}(1+m\theta)\left(m+1+\frac{1}{\theta^{n+1}}\right) + \frac{2}{\theta} \\
&\leq (1-\theta_1)^{m+1}(1+m\theta_2)\left(m+1+\frac{1}{\theta_1^{n+1}}\right) + \frac{2}{\theta_1}
\end{aligned}$$

Donc l'inégalité $\mathbb{E}_{2,\gamma_2,\theta}(T_{\text{ERREUR}}) > \mathbb{E}_{1,\gamma_1,\theta}(T_{\text{ERREUR}})$, pour tout $\theta \in [\theta_1, \theta_2]$ est vraie dans les domaines (m,n) suivants :

— à $n > \frac{2}{\theta_1} \times \frac{\theta_2}{1-\theta_2}$ fixé, l'inégalité est vraie pour m assez grand.

— si $n \to \infty$ et $m \to \infty$, alors l'inégalité est vraie si $\frac{(1-\theta_1)^{m+1}}{\theta_1^{n+1}} < n$ en ordre de grandeur, donc si $m+1 \geq (n+1)\frac{\log(1-\theta_1)}{\log(\theta_1)}$ par exemple.

Malheureusement, les bornes naïves présentées ici ne sont pas assez précises pour conclure lorsque $n \to \infty$ et m reste borné et dans les autres cas, mais il parait assez raisonnable de conjecturer qu'en fait, si on se donne deux bornes $0 < \theta_1 < \theta_2 < 1$, alors on peut trouver $m_0, n_0 \in \mathbb{N}$, alors pour $m, n \geq m_0, n_0$, $\mathbb{E}_{2,\gamma_2,\theta}(T_{\text{ERREUR}}) > \mathbb{E}_{1,\gamma_1,\theta}(T_{\text{ERREUR}})$, pour tout $\theta \in [\theta_1, \theta_2]$, mais il faudrait avoir des bornes plus précises, ce qui n'est pas vraiment le but de cette section (la mauvaise borne semble être l'utilisation de la borne 9 qui est très brutale).

Finissons par mentionner pourquoi l'inégalité $\mathbb{E}_{2,\gamma_2,\theta}(T_{\text{ERREUR}}) > \mathbb{E}_{1,\gamma_1,\theta}(T_{\text{ERREUR}})$ implique $\mathbb{E}_{2,\gamma,\theta}(T_{\text{ERREUR}}) > \mathbb{E}_{1,\gamma,\theta}(T_{\text{ERREUR}})$ pour tout γ état du système. Cela est en fait essentiellement dû au fait que les deux modèles ne diffèrent que d'une transition : on écrit alors en posant $\gamma = \begin{matrix} * & \cdots & * & \frac{m-2}{m} \\ * & 1 \end{matrix}$

$$\begin{aligned}
\mathbb{E}_{2,\kappa,\theta}(T_{\text{ERREUR}}) &= \sum_{x_1,\ldots,x_k \text{ états}} Q_2(\kappa,x_1)\ldots Q_2(x_{k-1},x_k)Q_2(x_k,\text{ERREUR})k \\
&= \sum_{\substack{x_1,\ldots,x_k\text{états} \\ \forall i\leq k-2,(x_i,x_{i+1})\neq(\gamma,\gamma_2),\text{et}(x_{k-1},x_k)=(\gamma,\gamma_2)}} Q_2(\kappa,x_1)\ldots Q_2(x_{k-1},x_k)(k+\mathbb{E}_{2,\gamma_2,\theta}(T_{\text{ERREUR}})) \\
&\qquad + \sum_{\substack{x_1,\ldots,x_k\text{états} \\ \forall i\leq k-1,(x_i,x_{i+1})\neq(\gamma,\gamma_2)}} Q_2(\kappa,x_1)\ldots Q_2(x_{k-1},x_k)Q_2(x_k,\text{ERREUR})k \\
&= \sum_{\substack{x_1,\ldots,x_k\text{états} \\ \forall i\leq k-2,(x_i,x_{i+1})\neq(\gamma,\gamma_1),\text{et}(x_{k-1},x_k)=(\gamma,\gamma_1)}} Q_1(\kappa,x_1)\ldots Q_1(x_{k-1},x_k)(k+\mathbb{E}_{2,\gamma_2,\theta}(T_{\text{ERREUR}})) \\
&\qquad + \sum_{\substack{x_1,\ldots,x_k\text{états} \\ \forall i\leq k-1,(x_i,x_{i+1})\neq(\gamma,\gamma_1)}} Q_1(\kappa,x_1)\ldots Q_1(x_{k-1},x_k)Q_1(x_k,\text{ERREUR})k
\end{aligned}$$

$$
\begin{aligned}
&> \sum_{\substack{x_1,\dots,x_k \text{états} \\ \forall i \leq k-2, (x_i,x_{i+1}) \neq (\gamma,\gamma_1), \text{et}(x_{k-1},x_k)=(\gamma,\gamma_1)}} Q_1(\kappa,x_1)\dots Q_1(x_{k-1},x_k)(k+\mathbb{E}_{1,\gamma_1,\theta}(T_{\text{ERREUR}})) \\
&\qquad + \sum_{\substack{x_1,\dots,x_k \text{états} \\ \forall i \leq k-1, (x_i,x_{i+1}) \neq (\gamma,\gamma_1)}} Q_1(\kappa,x_1)\dots Q_1(x_{k-1},x_k)Q_1(x_k,\text{ERREUR})k \\
&= \mathbb{E}_{1,\kappa,\theta}(T_{\text{ERREUR}})
\end{aligned}
$$

car pour la troisième égalité il n'y a que la transition $\gamma \to \gamma_2$ qui change en $\gamma \to \gamma_1$.

Remarque. On pourrait alors penser faire tendre $m \to \infty$ pour avoir un modèle avec des charges presque continues, et être donc continuellement dans le régime moyenne fréquence, mais il ne faut pas oublier de « scaler » θ aussi : si on augmente les capacités de charge de 2 par exemple ($m \to 2m$), alors puisque les voitures viennent au même moment mais que les tenders mettent une durée 2 fois plus grande à se recharger dans le modèle, le θ_0 qui convenait pour m se transforme presque en $\frac{\theta_0}{2}$ dans le modèle à $2m$ niveaux de charge selon le rapport entre les paramètres réels et ceux du modèle donné par l'équation 1.

5.4 Conclusions finales

Dans la section 2, on a montré que l'algorithme glouton était « faiblement » optimal. Dans cette section, on a réfuté le fait que l'algorithme glouton soit « fortement » optimal. En particulier, on a montré que changer quelques transitions de l'algorithme glouton pouvait le rendre meilleur dans des fréquences moyennes d'arrivée des voitures « modérées ». Ce qu'on a montré, c'est qu'il y a en fait compétition entre la longueur d'une file et le niveau de charge du dernier tender de celle-ci, dans des fréquences moyennes d'arrivée des voitures « modérées ».

Nous laissons à une étude plus détaillée le soin de déterminer jusque quelles transitions exactement il faut changer pour obtenir un algorithme totalement optimal. Nous nous contenterons de dire que lorsque la fréquences moyennes d'arrivée des voitures est « modérée », on gagne à changer quelques transitions en tenant compte de cette compétition, sans trop s'éloigner de l'algorithme glouton.

Nous nous sommes restreins à des modèles relativement simples pour avoir la possibilité de montrer des phénomènes intéressants, mais on peut bien sûr imaginer des modèles plus complexes dans lesquels on accepte des temps d'attente par exemple. Mais ces changements seraient finalement minimes et on peut conjecturer qu'une adaptation de l'algorithme glouton serait assez probablement assez optimal, et que le modifier légèrement de la même façon que ce qui a été fait (en prenant en compte la compétition longueur de la file - niveau de charge du dernier tender), amènerait à un algorithme de répartition encore plus optimal.

5.5 Implémentation

Au vu de l'étude précédente, l'algorithme glouton semble donc être déjà un algorithme très optimal. Pour avoir un algorithme plus optimal, on peut alors imaginer changer l'algorithme de répartition selon la fréquence moyenne d'arrivée des voitures.
Aux heures de pointe, la fréquence d'arrivée des voitures serait alors modérée (on ne veut pas qu'elle soit forte car sinon on va se retrouver dans tous les cas avec une file d'attente des voitures de plus en plus longue et un mécontentement des clients très important, et il apparaît alors favorable de mettre un nombre de files et de tenders dans les stations approprié de manière à garder une fréquence d'arrivée des

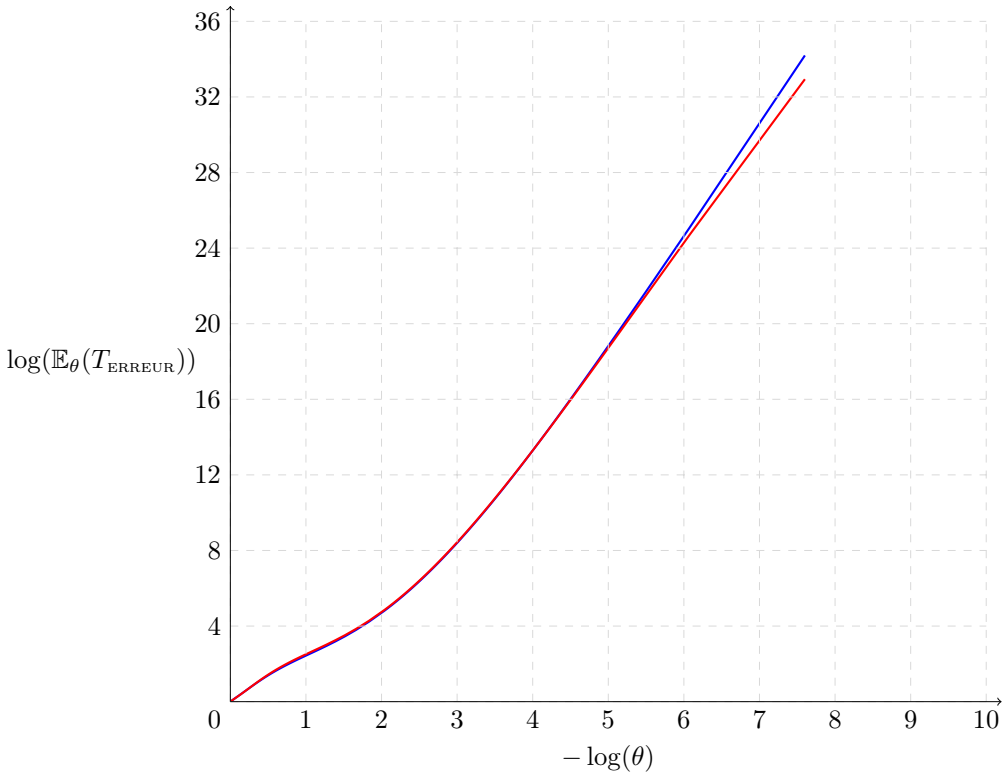

FIGURE 5 – Graphe en échelle logarithmique de la fonction $\theta \mapsto \mathbb{E}_\theta(T_{\text{ERREUR}})$ avec 2 files, 7 tenders et 11 niveaux de charge. La légende est la même que pour les figures de la première section : la courbe bleue correspond à l'algorithme glouton (premier modèle) et la rouge correspond au second modèle. On retrouve alors la domination de l'algorithme glouton (premier modèle) sur le second modèle pour les faibles valeurs de θ, et on avec de bons yeux, on aperçoit la domination du second algorithme sur le premier pour des valeurs de θ distantes de 0 et 1.

voitures modérée par rapport au nombre de files et de tenders). Alors, il est plus favorable de modifier l'algorithme glouton en prenant en compte une importance un peu plus grande de la taille des files que du niveau de charge du dernier tender de la file, que lorsque la fréquence d'arrivée des voitures est faible. On s'éloigne alors quelque peu de l'algorithme glouton (mais seulement quelque peu, il ne faut pas trop s'y éloigner non plus au risque de perdre l'optimalité soulignée dans la partie 2).
Aux heures creuses, on se rapproche en revanche de l'algorithme glouton, qui sera plus adapté.

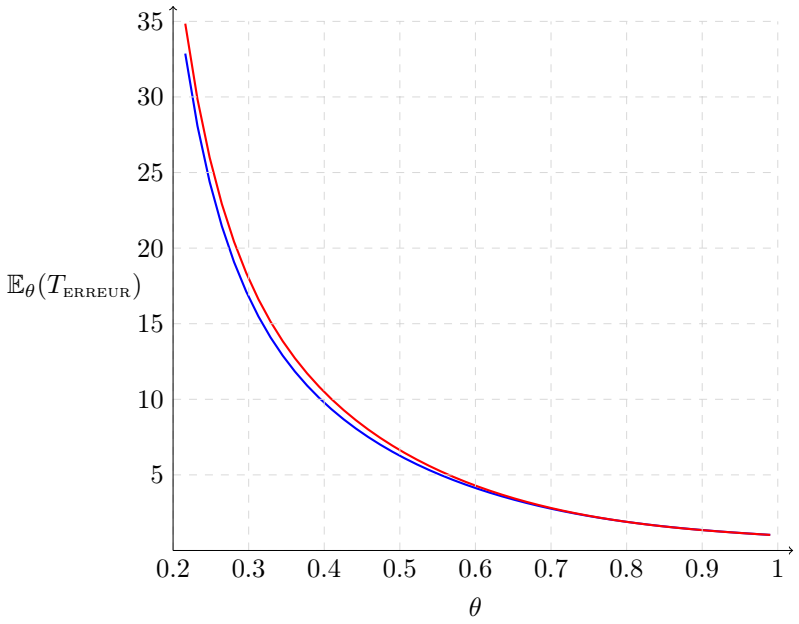

FIGURE 6 – Graphe de la fonction $\theta \mapsto \mathbb{E}_\theta(T_{\text{ERREUR}})$ avec 2 files, 7 tenders et 11 niveaux de charge. La légende est la même que dans la figure précédente. Cette figure montre que lorsque θ est éloigné de 0 et 1, le second modèle est meilleur que le premier.

6 Méthodes numériques

6.1 Gérer le compromis longueur/charge : la méthode des indices

On rappelle que l'algorithme glouton pour le rangement choisit les files les moins chargées, et, à même niveau de charge, les plus courtes, pour ranger les tenders. On peut voir cet algorithme dans un cadre plus général où on associe à chaque file un indice et où on choisit la file dont l'indice est le plus faible pour y ranger un tender.

Dans ce cadre plus général, on considère pour indice une fonction $f(l, c)$ où l désigne la longueur d'une file et c le niveau de charge de la file (c'est-à-dire du dernier tender ajouté à la file). Pour l'algorithme glouton, on prend par exemple

$$f(l, c) = (N + 1)c + l$$

où N est le nombre total de tenders. Ainsi, entre deux files, celle dont l'indice sera le plus élevé sera la plus chargée ; à même niveau de charge, celle dont l'indice sera le plus faible sera la plus courte.

En fait, on écrit plutôt $f(l, c, \lambda)$ où $\lambda \in \Lambda \subset \mathbb{R}^k$, $k \in \mathbb{N}$ est un paramètre à ajuster. On obtient ainsi des classes d'algorithmes données par les $f(\cdot, \cdot, \lambda)$, $\lambda \in \Lambda$ qu'on peut alors optimiser en λ pour adapter l'algorithme aux situations concrètes qui se présentent. On suggère dans la suite quelques classes d'algorithmes pour nourrir la réflexion future.

Nous devons ici souligner que ces classes d'algorithmes sont simplistes. D'autres paramètres pourraient être pris en compte et apporter des améliorations. Cependant, elles sont suffisantes pour mettre en place un compromis longueur/charge, qu'on peut calibrer à l'aide de λ.
Néanmoins, si des améliorations à ces algorithmes devaient être apportées, le plus important des paramètres additionnels à considérer est le nombre de tenders dans les files plus chargées. En effet, intuitive-

ment, si on a déjà un grand nombre de tenders chargés (ou plus chargés que ceux des files qu'on compare), on sera plus facilement en mesure de servir la demande future. On prend donc moins de risques en privilégiant de laisser une file plus longue mais moins chargée se remplir. Il s'agit donc d'un paramètre important.

Propriétés désirables pour les indices :
Les fonctions $f(l, c, \lambda)$ doivent être croissantes en c et en λ : on veut mettre en priorité les tenders dans les files les moins longues et les moins chargées. Ces deux propriétés combinées assurent qu'on ne bloquera jamais une file plus longue et plus chargée au détriment d'une autre, plus courte et moins chargée.

Puisque qu'on ne s'intéresse qu'à l'ordre dans les indices des files, on peut composer ces fonctions par n'importe quelle fonction croissante : le même algorithme peut donc s'exprimer sous différentes formes.

Enfin, quant au niveau de charge, trois logiques s'opposent. D'une part, on peut se dire que la seule chose qui importe quand on compare le niveau de charge de deux files est la différence entre les niveaux de charge. En effet, cela détermine le temps additionnel, gagné ou perdu selon la file dans laquelle on choisit de ranger le tender, pour obtenir les tenders chargés. Cette logique se tient bien si les deux files considérées sont les seules en station.
Cependant, comme souligné précédemment, le nombre de tenders en station plus chargés que les files que l'on compare joue aussi : s'il y en a peu, privilégier une file plus longue mais moins chargée est risqué. Le nombre de tenders plus chargés n'est pas pris en compte dans $f(l, c, \lambda)$, néanmoins, intuitivement, si les files que l'on considère sont peu chargées il est plus probable que les autres files soient plus chargées. Ainsi, selon cette logique, favoriser des files plus longues au détriment de files plus chargées est plus prudent pour les files les moins chargées.
Finalement, si l'on doit choisir dans laquelle de deux files ranger un tender, on peut se dire au contraire que plus ces files sont chargées, plus l'investissement en charge déjà réalisé sur ces files est important. En sacrifiant une file plus longue au profit d'une file plus chargée, la perte nette en charge totale est plus marquée si les files étaient très chargées. De plus, il n'y a dans tous les cas plus longtemps à attendre pour que ces files soient complètement chargées, ainsi, autant ranger le tender dans la file la plus courte et récolter un plus grand nombre de tenders chargés. D'une part, la récompense pour cette prise de risque advient d'autant plus rapidement que les files étaient très chargées. D'autre part, en concentrant le jeu du compromis longueur/charge sur les files les plus chargées, on s'assure que les files dans lesquelles on vient de ranger un tender, et qui sont donc très déchargées, restent prioritaires (en l'absence de files vides) pour le rangement de tenders. On évite ainsi de ranger les tenders sans cesse dans des files différentes et de ne pas laisser aux files le temps de se charger.

Dans le premier cas, le gain d'un niveau de charge résulte en un gain constant dans la valeur d'une file. On cherche des fonctions de la forme :

$$f(l, c, \lambda) = c + u(l) \qquad \text{où } u \text{ est une fonction de valeur (croissante) pour la longueur des files}$$

Dans le second cas, le gain d'un tender justifie la perte d'un niveau de charge d'autant plus élevé que le niveau de charge initial est faible. Autrement dit, le gain d'un niveau de charge résulte en un gain d'autant plus important dans la valeur d'une file que le niveau de charge initial était élevé. On cherche des fonctions vérifiant :

$$\forall l \in \mathbb{N},\ \lambda \in \Lambda,\ f(l, \cdot, \lambda) \text{ est convexe}$$

Dans le dernier cas, le gain d'un tender justifie la perte d'un niveau de charge d'autant plus élevé que le niveau de charge initial est élevé. Autrement dit, le gain d'un niveau de charge résulte en un gain d'autant plus important dans la valeur d'une file que le niveau de charge initial était faible. On cherche des fonctions vérifiant :

$$\forall l \in \mathbb{N},\ \lambda \in \Lambda,\ f(l, \cdot, \lambda) \text{ est concave}$$

Les résultats empiriques qui suivent montrent que les indices concaves en c sont les plus performants, et vont donc dans le sens de la dernière logique présentée.

6.2 Simulation numérique et choix des paramètres

Dans la suite, nous allons présenter différentes classes d'algorithmes basées sur différents indices, et évaluer empiriquement le choix optimal du paramètre λ pour chacune de ces classes. On pourra ainsi comparer les performances de chacune de ces classes à celles de l'algorithme glouton. Cela sera fait dans la section 6.3. Cette approche n'a pas pour but de donner un algorithme optimal dans l'absolu : le choix de la classe d'algorithme et du meilleur paramètre devra être calibré empiriquement aux paramètres des stations réelles. Cependant, cette approche nous permettra de construire une intuition sur le comportement de chacune des classes et de vérifier le bénéfice qu'elles peuvent apporter par rapport à l'algorithme glouton. Dans la section 6.4, on évaluera la robustesse de ces classes d'algorithmes à une mauvaise spécification des paramètres. Si on a calibré un algorithme pour la fréquence d'arrivée des clients estimée d'une station, mais que la fréquence d'arrivée réelle se révèle différente, cela nuit-il gravement aux performances de l'algorithme ?

On va choisir les paramètres de manière à se placer dans un cadre réaliste. Dans toute la section 6.3, on prendra $\tau = 2$ heures pour le temps mis par une file pour charger un tender, et $\rho = 5$ le nombre de clients par heure. On a donc en moyenne un client tous les $T = 0.2$ heures (soit toutes les 12 minutes). On conserve $m = 2$ niveaux de charge pour éviter une explosion du nombre d'états dans notre chaîne de Markov (qui aurait pour conséquence une explosion du temps de calcul). On considère une station avec $f = 3$ files et $N = 20$ tenders, de sorte qu'on est bien dans le cas :

$$2 = \tau \leq T\left(1 + N\frac{f-2}{f-1}\right) = \frac{11}{5}$$

où l'algorithme glouton serait capable de soutenir la demande sur le long terme, si cette dernière était parfaitement régulière.

Dans l'ensemble de la section qui suit, les degrés de charge prennent de valeurs entières, de 0 à m. On peut se ramener au cas où les valeurs prises sont $0, \frac{1}{m}, \ldots, 1$ en multipliant les indices par un facteur approprié, et en dérivant ainsi la nouvelle valeur des paramètres optimaux.

L'ensemble du code utilisé pour ces simulations est disponible sur GitHub, sous la forme de Jupyter Notebooks, au lien suivant. On réfère le lecteur au README pour la présentation des fichiers.

6.3 Suggestions d'algorithmes

6.3.1 Les indices linéaires

Si l'on est prêt à sacrifier un niveau de charge pour augmenter la longueur d'une file d'un nombre constant de tenders, on considère la classe :

$$f(c, l, \lambda) = \lambda c + l \qquad \text{où } \lambda \geq 0$$

Ici, λ désigne le nombre de tenders justifiant la perte d'un niveau de charge (pas nécessairement entier). Plus λ est grand, plus on se rapproche de l'algorithme glouton.
Pour un λ bien calibré, on obtient une amélioration mineure sur l'algorithme glouton.

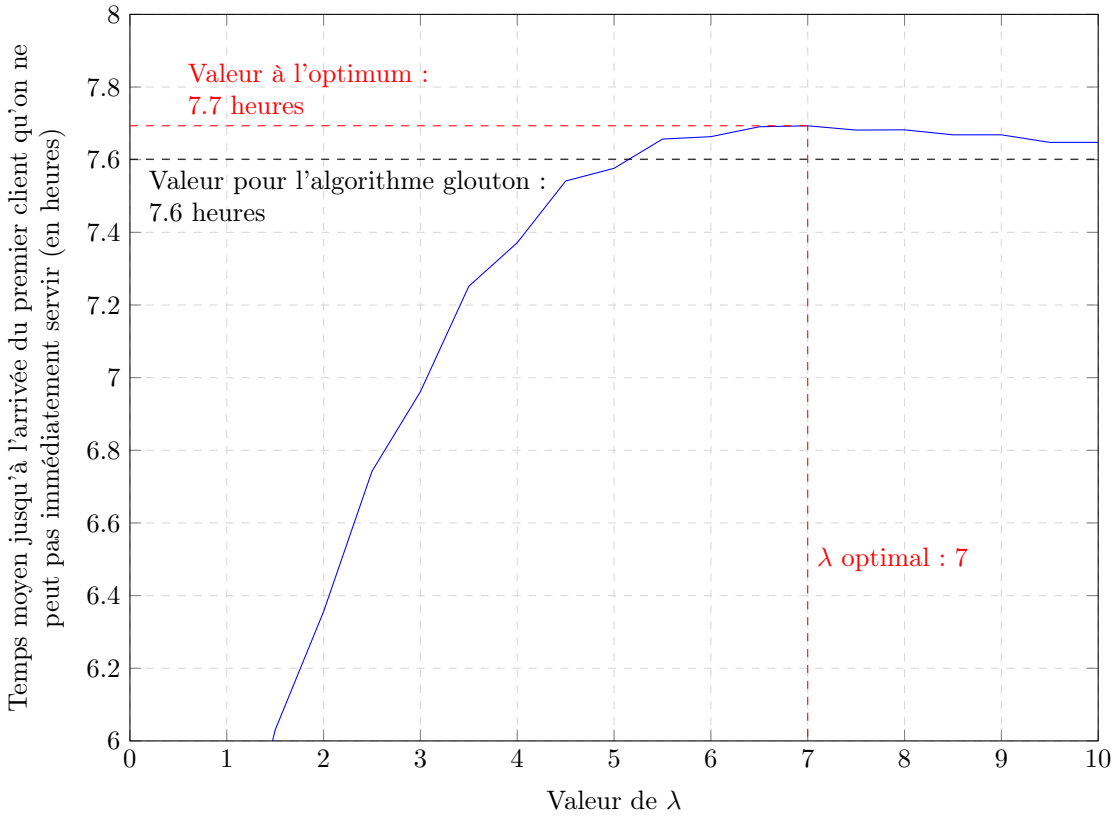

FIGURE 7 – Temps moyen, en heures, jusqu'à l'arrivée du premier client qu'on ne peut pas servir, pour la classe d'algorithme donnée par les indices linéaires de paramètre λ

6.3.2 Les indices logarithmiques

Si l'on est prêt à sacrifier un niveau de charge pour multiplier la longueur d'une file d'un facteur constant, on considère la classe :

$$f(c, l, \lambda) = c \log \lambda + \log l \qquad \text{où } \lambda > 1$$

Ici, on est prêt à sacrifier un niveau de charge dans une file pour multiplier sa longueur par un facteur $\lambda > 1$. Encore une fois, plus λ est grand et plus on se rapproche de l'algorithme glouton.
Pour un λ bien calibré, on obtient une amélioration mineure sur l'algorithme glouton.

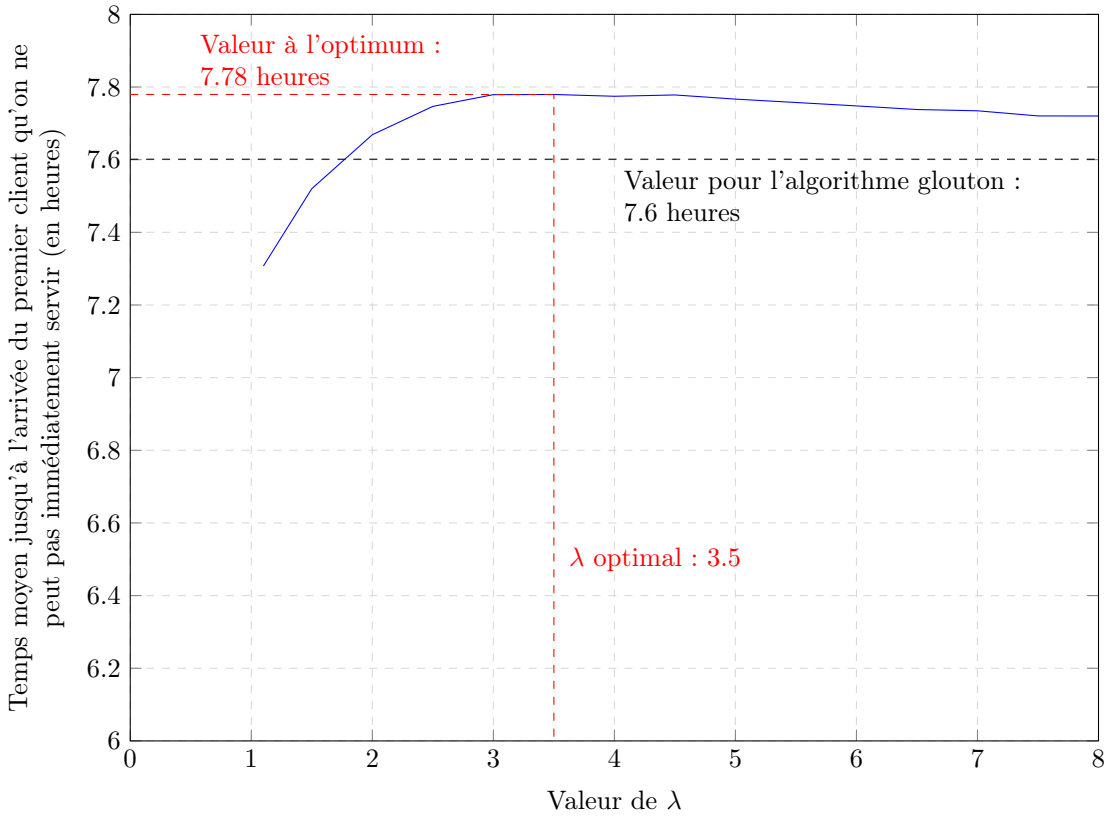

FIGURE 8 – Temps moyen, en heures, jusqu'à l'arrivée du premier client qu'on ne peut pas servir, pour la classe d'algorithme donnée par les indices logarithmiques de paramètre λ

6.3.3 Les indices multiplicatifs

Si l'on est prêt à diviser le niveau de charge d'une file d'un facteur x pour multiplier sa longueur par un facteur x^λ, on considère la classe :

$$f(c, l, \lambda) = cl^\lambda \qquad \text{où } \lambda > 0$$

En composant par log, on peut réécrire cette classe sous la forme :

$$f(c, l, \lambda) = \log c + \lambda \log l$$

On se rapproche de l'algorithme glouton en laissant λ tendre vers 0.
Pour un λ bien calibré, on obtient une amélioration mineure sur l'algorithme glouton.

Une classe d'indices similaire mais qui remplace le niveau de charge par le "niveau de décharge", c'est-à-dire le temps d'attente restant jusqu'à charge pleine, est donnée par les indices multiplicatifs inversés :

$$f(c, l, \lambda) = \frac{l^\lambda}{\tau - c} \qquad \text{où } \lambda > 0 \text{ ou}$$
$$f(c, l, \lambda) = \lambda \log l - \log(\tau - c)$$

Cette classe exploite la même intuition que la précédente, mais cette fois-ci on est prêt à multiplier par x le temps d'attente jusqu'à la charge complète de la file pour multiplier la longueur de la file par x^λ.

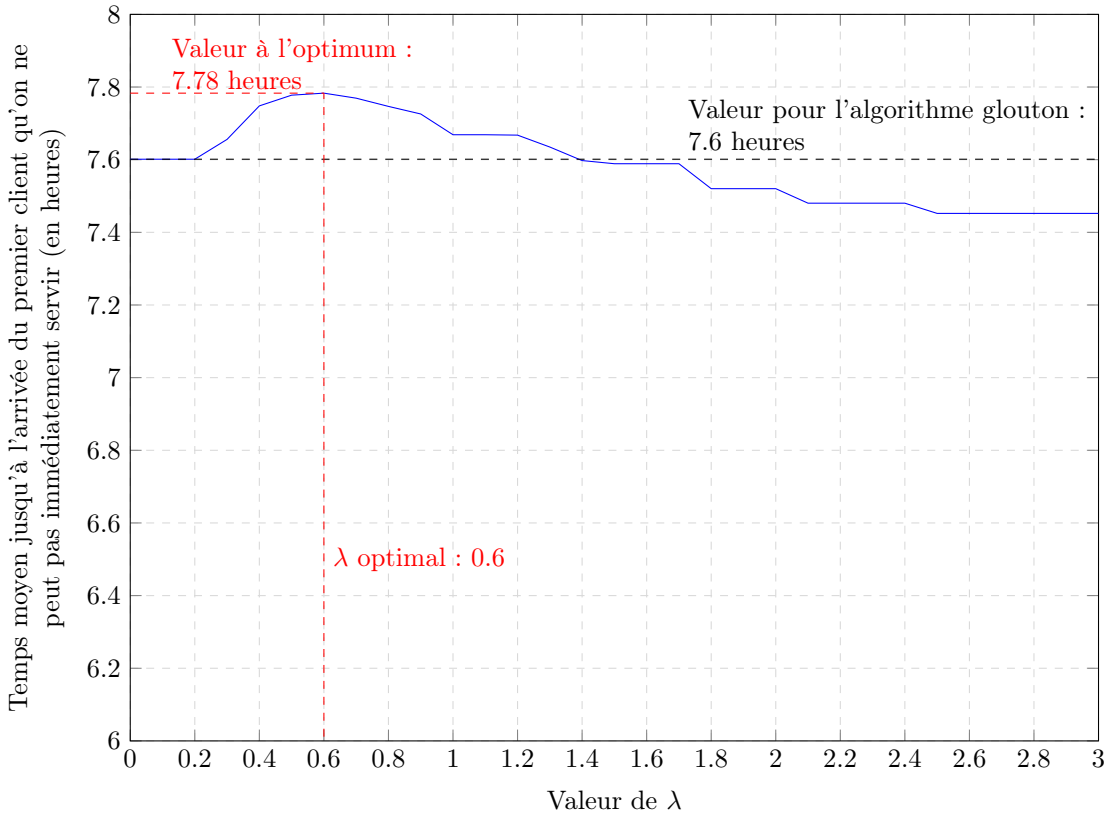

FIGURE 9 – Temps moyen, en heures, jusqu'à l'arrivée du premier client qu'on ne peut pas servir, pour la classe d'algorithme donnée par les indices multiplicatifs de paramètre λ

Cette fois, $f(\cdot, l, \lambda)$ est convexe pour chaque l et $\lambda > 0$, alors qu'avant $f(\cdot, l, \lambda)$ était concave.

Les indices multiplicatifs inversés ne sont pas plus performants que les indices multiplicatifs, en fait, ils ne sont plus performants que l'algorithme glouton pour aucun paramètre.

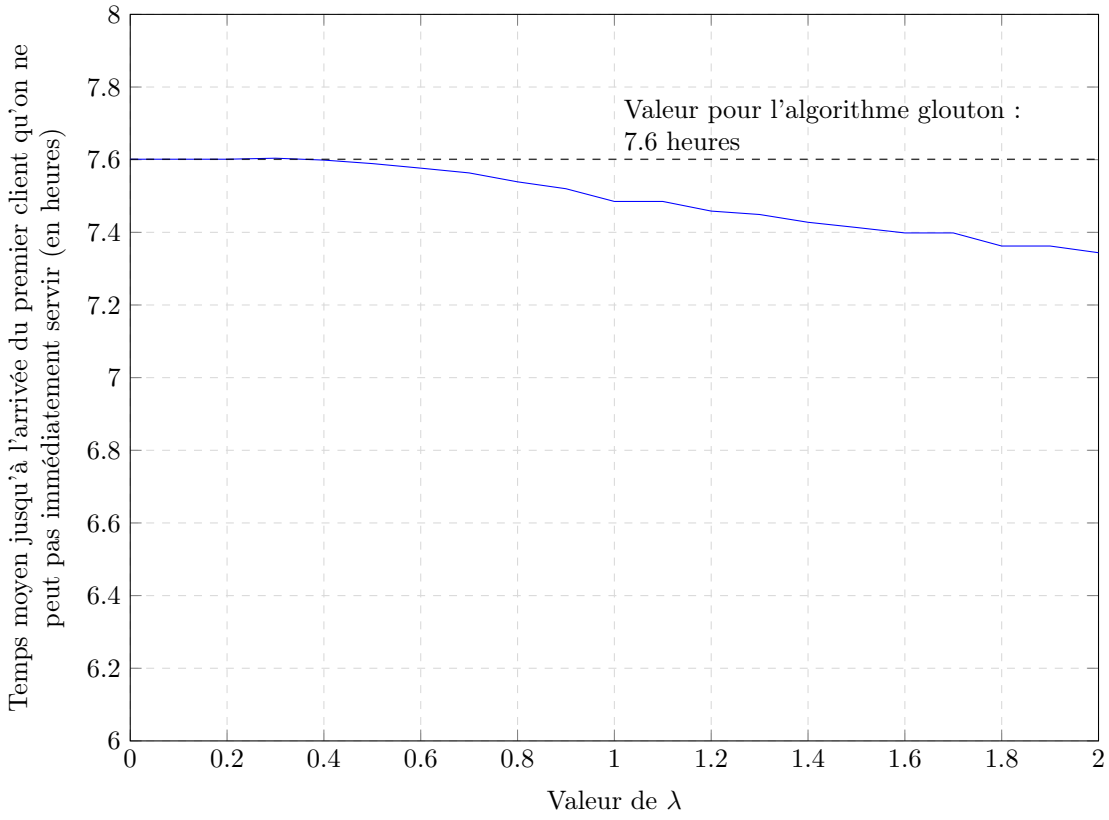

FIGURE 10 – Temps moyen, en heures, jusqu'à l'arrivée du premier client qu'on ne peut pas servir, pour la classe d'algorithme donnée par les indices multiplicatifs inversés de paramètre λ

6.3.4 Les indices puissance

On peut modifier les indices linéaires pour intégrer un caractère concave ou convexe en c :

$$f(c, l, \lambda) = c^{\lambda_0} + \lambda_1 l \qquad \text{où } \lambda_0, \lambda_1 > 0$$

$f(\cdot, l, \lambda)$ est, pour tout l et tout λ_1, concave en c lorsque $\lambda_0 \leq 1$ est convexe en c et $\lambda_0 \geq 1$. On se rapproche de l'algorithme glouton en laissant λ_1 tendre vers 0.
Pour un λ bien calibré, on obtient une amélioration mineure sur l'algorithme glouton.

Ici encore, l'indice optimal est obtenu pour $\lambda_0 = 3 > 1$ qui correspond à un indice concave en c.
En fait, cet algorithme ne permet un gain d'efficacité par rapport à l'algorithme glouton que lorsque $\lambda_0 \leq 1$, c'est-à-dire lorsque les indices sont concaves en c.

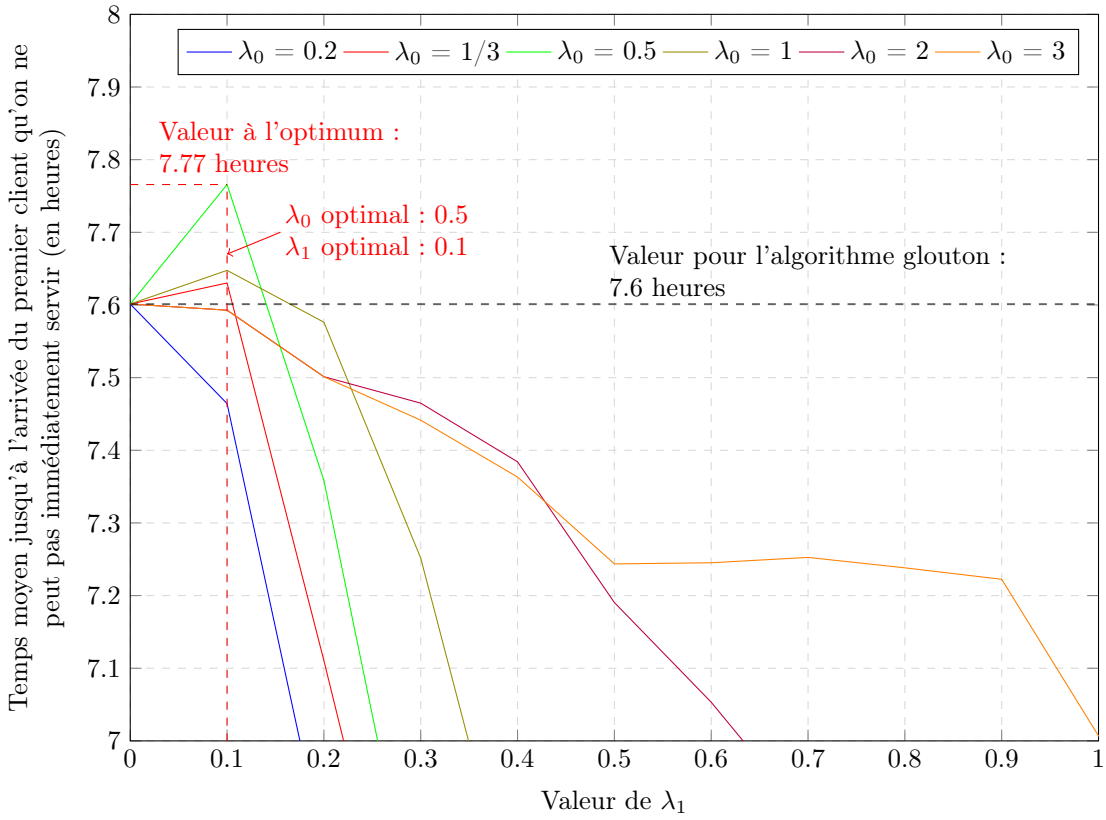

FIGURE 11 – Temps moyen, en heures, jusqu'à l'arrivée du premier client qu'on ne peut pas servir, pour la classe d'algorithme donnée par les indices puissance de paramètre λ

Cette première version des indices puissance-log considère c^{λ_0}, c'est-à-dire une fonction du degré de charge accompli des tenders. Il peut être plus pertinent de considérer à la place le degré de charge manquant des tenders. On considère alors les indices :

$$f(c, l, \lambda) = \lambda_1 \log l - (\tau - c)^{\lambda_0} \qquad \text{où } \lambda_0, \lambda_1 > 0$$

Ces indices sont convexes en c lorsque $\lambda_0 \leq 1$ et concaves en c lorsque $\lambda_0 \geq 1$.

Quand il n'y a que deux degrés de charge comme ici, les deux classes sont équivalentes, mais si le nombre de degrés de charge est plus important (ou si le niveau de charge est continu), elles sont très différentes.

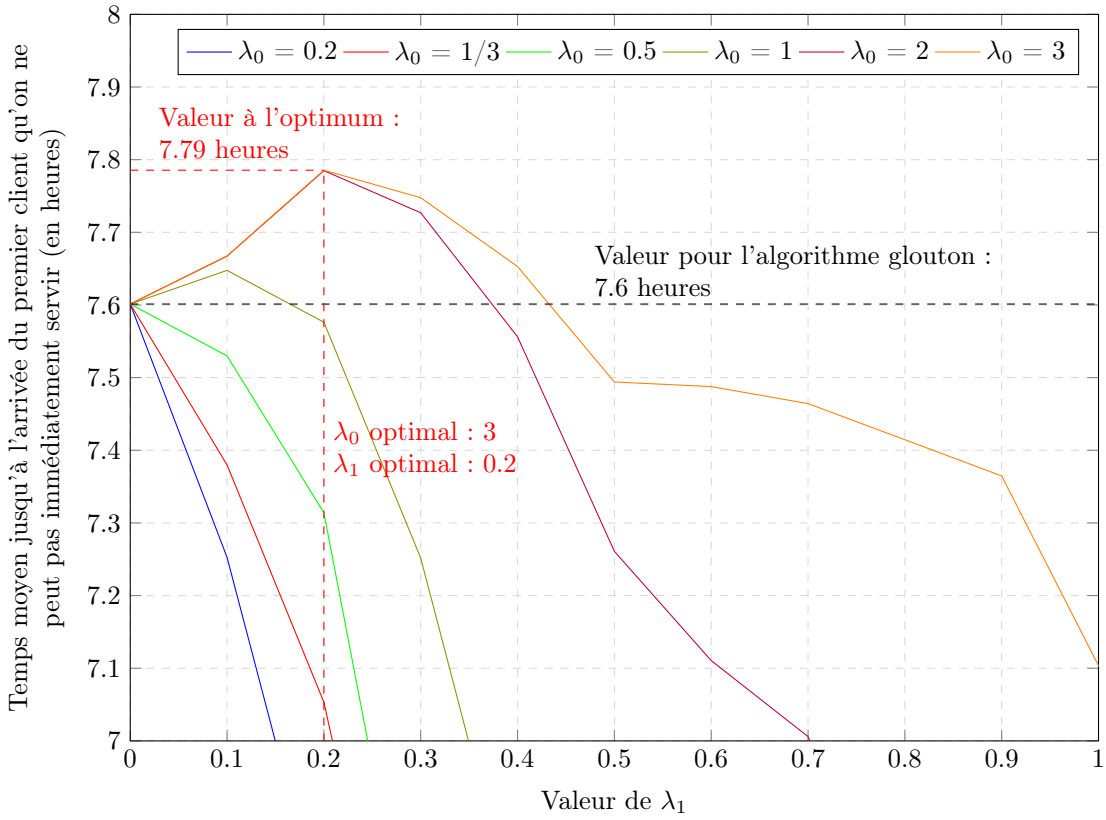

FIGURE 12 – Temps moyen, en heures, jusqu'à l'arrivée du premier client qu'on ne peut pas servir, pour la classe d'algorithme donnée par les indices puissance inversés de paramètre λ

6.3.5 Les indices puissance-log

On peut aussi modifier les indices logarithmiques pour intégrer un caractère concave ou convexe en c :

$$f(c, l, \lambda) = c^{\lambda_0} + \lambda_1 \log l \qquad \text{où } \lambda_0, \lambda_1 > 0$$

$f(\cdot, l, \lambda)$ est, pour tout l et tout λ_1, concave en c lorsque $\lambda_0 \leq 1$ est convexe en c et $\lambda_0 \geq 1$. On se rapproche de l'algorithme glouton en laissant λ_1 tendre vers 0.
Cet algorithme ne permet un gain d'efficacité par rapport à l'algorithme glouton que lorsque $\lambda_0 < 1$, c'est-à-dire lorsque les indices sont concaves en c.

Cette première version des indices puissance-log considère c^{λ_0}, c'est-à-dire le degré de charge accompli des tenders. Il peut être plus pertinent de considérer à la place le degré de charge manquant des tenders. On considère alors les indices :

$$f(c, l, \lambda) = \lambda_1 \log l - (\tau - c)^{\lambda_0} \qquad \text{où } \lambda_0, \lambda_1 > 0$$

Ces indices sont convexes en c lorsque $\lambda_0 \leq 1$ et concaves en c lorsque $\lambda_0 \geq 1$.

Quand il n'y a que deux degrés de charge comme ici, les deux classes sont équivalentes, mais si le nombre de degrés de charge est plus important (ou si le niveau de charge est continu), elles sont très différentes.

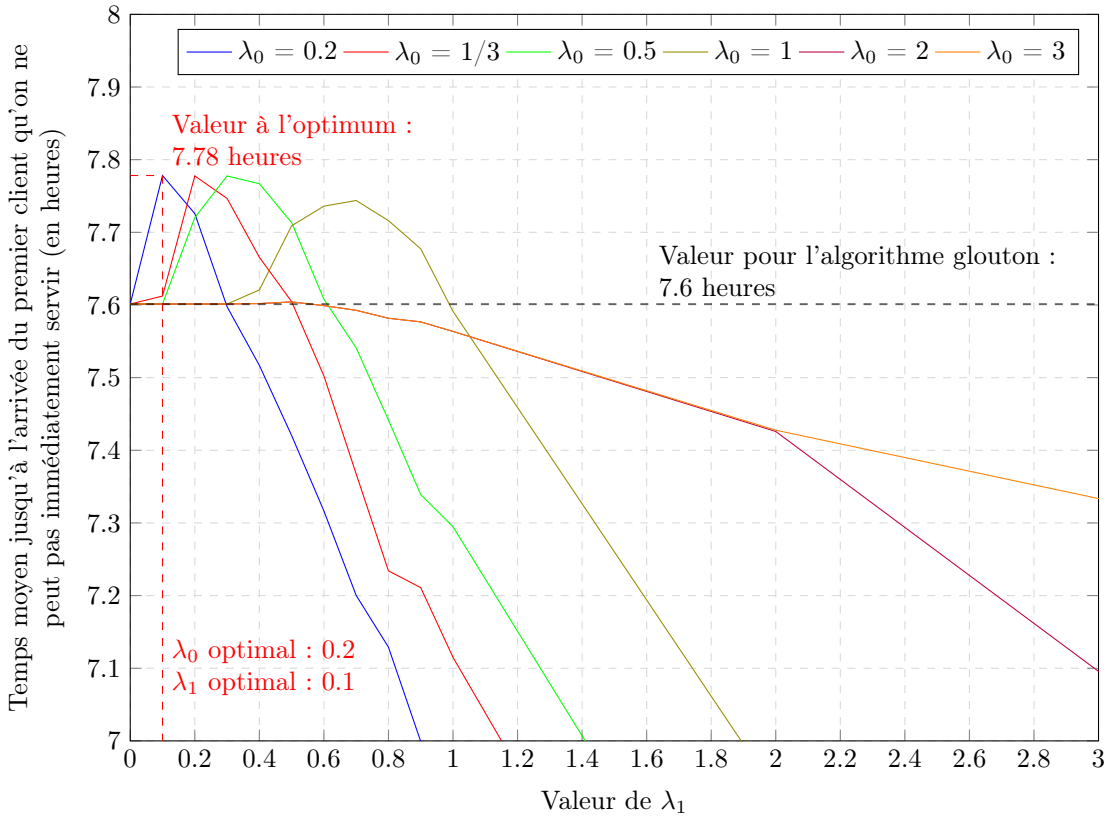

FIGURE 13 – Temps moyen, en heures, jusqu'à l'arrivée du premier client qu'on ne peut pas servir, pour la classe d'algorithme donnée par les indices puissance-logarithme de paramètre λ

Ici encore, l'indice optimal est obtenu pour $\lambda_0 = 3 > 1$ qui correspond à un indice concave en c.

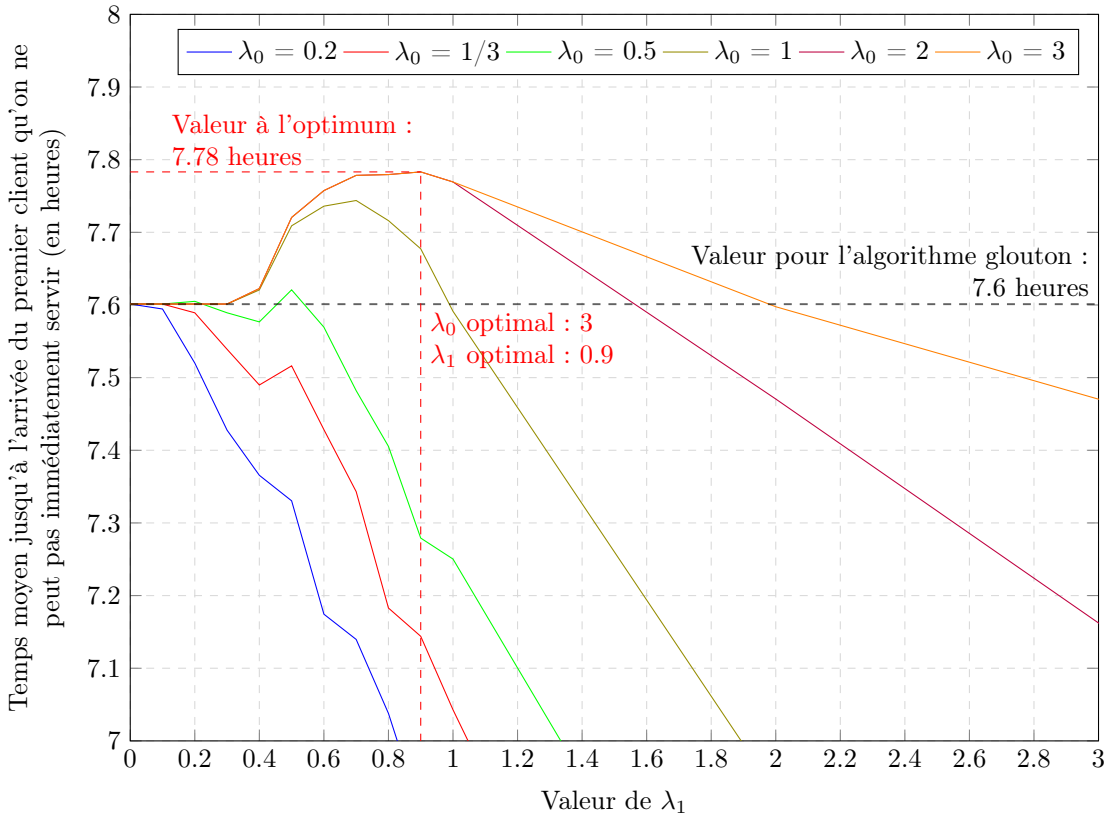

FIGURE 14 – Temps moyen, en heures, jusqu'à l'arrivée du premier client qu'on ne peut pas servir, pour la classe d'algorithme donnée par les indices puissance-logarithme inversés de paramètre λ

6.4 Robustesse des algorithmes face à une mauvaise spécification des paramètres

Dans cette dernière section, on s'intéresse à la sensibilité de nos algorithmes par rapport à la fréquence d'arrivée des voitures clientes. En effet, en pratique, on ne connaît pas en avance cette fréquence et on ne peut que l'estimer. On s'attend donc à ce qu'il y ait une différence entre la valeur de θ pour laquelle on optimise notre algorithme, et sa valeur 'réelle'. Si nos algorithmes sont très sensibles à une telle différence, l'algorithme optimisé pourrait avoir des performances pires que celles de l'algorithme glouton, alors que le modèle pour la valeur estimée de θ prédisait une amélioration. Au contraire, si ces algorithmes sont peu sensibles à une telle différence, alors on peut les utiliser en toute sérénité.
Pour mesurer la robustesse de nos algorithmes, on conserve, pour chacune des classes, la valeur de λ optimale pour $N = 20$, $\theta = \frac{5}{6}$, $f = 3$, et $d = 2$. En revanche, on fait désormais varier la *véritable* valeur du flux de clients par heure $\rho = \frac{\theta}{1-\theta}$, et on compare ces algorithmes pour voir si leurs performances continuent d'être meilleures que pour l'algorithme glouton.
On constate que les algorithmes optimisés pour un flux $\rho = 5$ restent plus performants que l'algorithme glouton pour un flux réel $2 \leq \rho_{\text{réel}} \leq 10$, à l'exception des algorithmes multiplicatifs inversés (qui ne sont jamais plus performants que l'algorithme glouton), et des algorithmes puissance-log (qui ne restent plus performants que l'algorithme glouton que pour $4 \leq \rho_{\text{réel}} \leq 10$). À l'exception de ces deux derniers algorithmes, nos algorithmes sont donc relativement robustes à une erreur dans l'estimation du flux moyen de clients.
Parmi ces algorithmes, ce sont les algorithmes multiplicatifs et puissance-log inversés qui permettent les

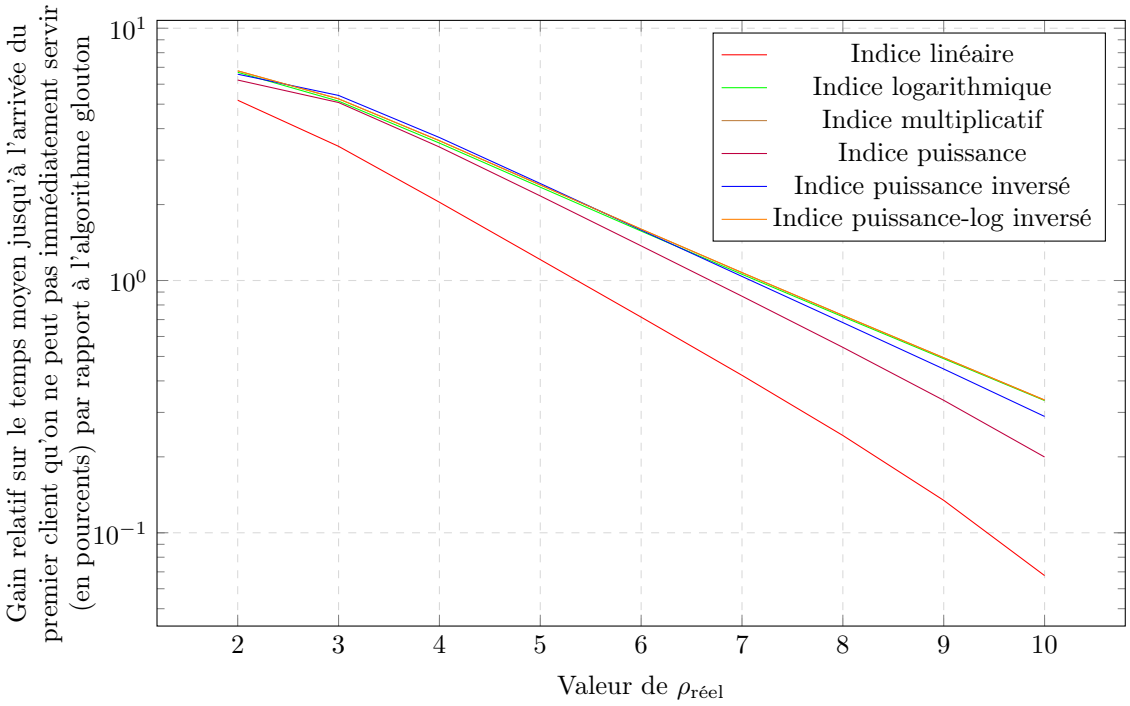

FIGURE 15 – Gain relatif sur les performances de chacune des classes d'algorithmes relativement à l'algorithme glouton.
Ces algorithmes ont été optimisés pour un flux de $\rho = 5$ clients par heure mais sont confrontés à un flux de client $\rho_{\text{réel}}$.

gains d'efficacité les plus élevés sur l'ensemble des valeurs de $\rho_{\text{réel}}$.

7 Limites de l'étude et pistes de réflexion pour le futur

Pour terminer ce rapport, nous proposons des pistes de réflexion sur des points que nous n'avons pas traités mais pertinents pour l'étude.

Premièrement, pour simplifier les modèles, on a supposé à partir de la section 3 qui introduit les modèles probabilistes, que les voitures rendaient un tender en même temps qu'elles en prenaient. On a donc imposé que la fréquence de retour des tenders en station soit égale à celle de demande des tenders. On peut cependant imaginer que ce ne soit pas le cas : le matin d'un jour de départ en vacances, la demande en tenders peut être élevée sans que beaucoup de tenders ne reviennent en station, les voitures n'ayant pas encore effectué le trajet souhaité ; le soir à l'inverse, il y aurait un important retour de tenders mais une demande faible. Une suite possible à ce rapport serait alors d'étudier des modèles dans lesquels les fréquences de retour et de demande des tenders sont décorrélées.
Notre intuition est que les résultats présentés dans ce rapport s'appliqueraient bien à des stations "relais", à mi-chemin le long d'un grand axe, mais qu'une étude des modèles où l'arrivée des tenders est décorrélée de leur départ serait pertinente pour les stations proches des grandes villes, qui se prêtent particulièrement aux scénarios de "grands départs" et de "grands retours".
Dans ces dernières stations, lorsqu'il y a une demande importante en tender et peu de retours, on s'attend à ce que l'algorithme glouton soit moins performant car on doit pouvoir satisfaire la demande de beaucoup de voitures. En revanche, lorsque le retour de tender est conséquent mais que la demande en tender est faible, on s'attend à ce que l'algorithme glouton soit efficace. En effet, on souhaite pouvoir servir rapidement les quelques clients qui viennent, et on veut donc éviter que les tenders qui arrivent en station gênent les files les plus chargées.

Deuxièmement, toujours à partir de la section 3, on a évalué la qualité des algorithmes considérés par la durée moyenne pendant laquelle ils permettent de servir instantanément tous les clients. On a donc ignoré tout ce qui se déroulait après l'arrivée du premier client ne pouvant être servi immédiatement. Cependant, imposer un temps d'attente à certains clients peut être toléré si ce temps reste court. On peut assimiler le mécontentement d'un client à une fonction de son temps d'attente. Une suite possible de l'étude serait alors de comparer les algorithmes selon le mécontentement moyen qu'ils induisent chez les clients : on essaierait de trouver l'algorithme qui engendre un mécontentement moyen minimal.
Notre intuition est qu'une file d'attente courte implique un faible temps d'attente et donc une quantité faible de mécontentement. Pour étudier ce modèle, on pourrait donc démultiplier tous les états de la chaîne de Markov utilisée dans ce rapport en plusieurs états, chacun représentant la même organisation des tenders dans la station mais donnant en plus la longueur de la file d'attente devant la station. Le modèle ainsi obtenu est toujours Markovien ; sous provision que la puissance de charge de la station soit capable d'absorber la demande moyenne, on s'attend à ce que la chaîne de Markov soit positive récurrente et qu'il y ait donc une distribution invariante sur ses états. On obtient alors une distribution invariante sur la longueur de la file d'attente, qu'on peut utiliser pour estimer le mécontentement moyen des clients. Déterminer algébriquement cette distribution invariante apparaît difficile, mais le théorème ergodique pour les chaînes de Markov en permet une estimation numérique aisée.
On ne s'attend cependant pas à ce que le comportement des modèles diffère grandement de ceux des modèles étudiés dans ce rapport. En effet, ces premiers modèles se ramènent aux seconds dans le cas d'un effet de seuil dans le mécontentement des clients : si les clients sont prêts à attendre dix minutes sans problème, mais pas une de plus, on peut utiliser les modèles présentés dans ce rapport en diminuant de dix minutes la durée de charge des tenders. Intuitivement, quand une voiture arrive, on lui "réserve" un tender qui sera chargé dans au plus dix minutes.

Finalement, pour les simulations numériques, nous avons implémenté une solution "exacte" pour estimer le temps moyen jusqu'à l'arrivée du premier client, consistant à résoudre un système d'équations. Le nombre de variables du système est le même que le nombre d'états ; quand le nombre de tenders, de files ou de niveaux de charge est grand, la résolution du système prend du temps. Implémenter une estimation numérique de ce temps moyen basée sur les méthodes de Monte-Carlo permettrait de faire des simulations plus rapides et donc de mieux simuler le comportement de stations plus grandes.

Références

[Gal06] Jean-François Le Gall. *Intégration, probabilités et processus aléatoires*. Septembre 2006. Disponible à `https://www.imo.universite-paris-saclay.fr/~jean-francois.le-gall/IPPA2.pdf`.