

---

## Des réseaux de capteurs sans fil au service des infrastructures routières

---

### Les communications sans fil : vers une réalité

D'une manière générale, TAPIOCA (chapitre 4) peut être utilisé sur n'importe quel support de communication. Toutefois, si les contrôleurs d'intersections sont parfois connectés entre eux par câble (p. ex., fibre optique), le lien entre les capteurs est plus vraisemblablement sans fil. S'affranchir de connexions câblées permet notamment de réduire le temps de déploiement, au même titre que les prix d'installation. Si la plupart des contributions utilisant une telle architecture pour gérer les feux de circulation considèrent un réseau de communication idéal, exempt de pertes ou de collisions, nous étudions, dans ce chapitre, les effets d'une situation plus réaliste. Tout comme dans les chapitres précédents, nous assumons le choix d'utiliser un réseau de capteurs sans fil basé sur le standard IEEE 802.15.4, particulièrement adapté aux dispositifs dont la consommation énergétique peut être un frein.

TAPIOCA repose sur une transmission régulière des véhicules comptabilisés, afin que, après un certain temps (si ce n'est immédiatement), le nœud de décision reçoive les informations. Dans les réseaux de capteurs sans fil, la perte de paquets due à des perturbations ou à des congestions peut toutefois causer de réels problèmes : les données peuvent être manquantes ou retardées et donc peuvent ne pas être à jour quand elles atteignent le nœud de décision. Compte tenu de ces facteurs, nous évaluons dans ce chapitre le comportement de TAPIOCA face à des contraintes de communication réelles. Nous supposons nous reposer sur l'architecture décrite par la figure 3.1 en section 3.2, car elle reste la plus communément utilisée.

## 5.1 Délais et problèmes de transmission

En nous basant sur une co-simulation – liant une simulation réseau à une simulation du trafic routier – nous évaluons l'architecture du réseau de capteurs sans fil dans la section 5.5 et montrons l'importance des retards qui peuvent être générés,

ainsi que des erreurs dans la comptabilisation des véhicules. En moyenne sur les simulations réalisées, le temps entre la génération et l'utilisation de mesures peut varier de 1 à 46 secondes. L'erreur de comptabilisation observée lorsque le nœud de décision calcule une nouvelle phase peut varier en fonction des retards de 6,6 % à 56,5 %. Cela signifie que le nombre de véhicules connus sur le nœud décision est potentiellement très éloigné du nombre de véhicules présents réellement, au même instant. Cela peut provoquer une perte importante dans le temps moyen d'attente des usagers. En effet, les décisions prises sur le nœud décision sont faussées par les retards de transmission. En conséquence, l'algorithme de TAPIOCA doit être modifié afin de le rendre résistant aux pertes et aux retards, dans une certaine mesure.

## 5. 2 Prédiction des données perdues

Nous cherchons tout d'abord à savoir si l'effet des pertes de paquets et des retards pourrait être atténué avec un mécanisme de prédiction des données. Ce mécanisme remplacerait les données manquantes par des valeurs interpolées en cas de besoin.

Les nœuds échangent périodiquement des comptabilisations de véhicules, qui reflètent des situations à des moments différents. Afin de mettre en place un tel mécanisme et permettre aux nœuds d'interpoler les valeurs manquantes, chaque nœud  $SN$  et  $DN$  maintient et transmet une estimation du taux de véhicules qu'il comptabilise.  $\Delta_k$  est la date à laquelle le  $k^{ieme}$  véhicule est détecté par un capteur. Le temps  $\tau_k^{(i)}$  qui sépare deux de ces événements sur le nœud  $i$  peut être approché en utilisant un filtre EWMA (*Exponentially Weighted Moving Average*,  $\alpha$  étant le paramètre de filtre) :

$$\tau_k^{(i)} = \alpha \cdot (\Delta_k - \Delta_{k-1}) + (1 - \alpha) \cdot \tau_{k-1}^{(i)},$$

Tout nœud  $SN$   $i$  peut alors maintenir un taux moyen d'arrivée  $\lambda_i = 1/\tau_k^{(i)}$  qu'il transmet avec des comptabilisations de véhicules réelles à son nœud  $DN$ . Tout nœud  $DN$   $j$  peut également maintenir un taux moyen de départ, filtré,  $\mu_j = 1/\tau_k^{(j)}$ , qu'il transmet à son agrégateur.

### 5. 2. 1 Prédiction sur le nœud $DN$

Les nœuds  $DN$  s'attendent à recevoir des données périodiquement des nœuds  $SN$ , chaque  $\Delta_{SN}$  secondes, qui devrait être inférieure à une durée de phase. L'effet de la valeur de  $\Delta_{SN}$  est étudié dans la section 5. 5.

Les nœuds fonctionnent de manière asynchrone et les décalages de l'horloge et les accès au support de communication peuvent ajouter des retards à la transmission du message. De ce fait, les nœuds  $DN$  définissent pour chaque nœud  $SN$  un *timer* (c.-à-d. une temporisation), dont l'expiration est programmée après  $\delta_{SN}$  secondes et qui est remis à zéro chaque fois qu'un message est reçu du nœud  $SN$ . Lorsque le timer expire, le nœud  $DN$  conclut qu'une donnée a été perdue.

Le nombre d'arrivées sur le nœud  $SN\ i$  à la date  $t$ ,  $\eta_i^A(t)$ , se déduit simplement de la valeur de ce paramètre à la date  $\tau_{SN_i}$  en ajoutant la différence de temps multipliée par le taux interpolé des arrivées,  $\lambda_i$  :

$$\eta_i^A(t) = \eta_i^A(\tau_{SN_i}) + (t - \tau_{SN_i}) \cdot \lambda_i .$$

Si  $\delta_{SN_i}$  est généralement plus grand que  $\Delta_{SN}$  pour faire face aux pertes de paquets et aux retards excessifs, il peut également être réglé à une valeur inférieure à  $\Delta_{SN}$ , afin de fournir des estimations intermédiaires sur le nombre de véhicules, en le lissant.

Lorsque  $\delta_{SN}$  expire ou quand un message est reçu d'un nœud  $SN\ i$ , le nœud  $DN\ j$  situé sur la même voie calcule le nombre total de véhicules présent sur cette voie. Il est noté  $\eta_j = \eta_i^A - \eta_j^D$  et est fonction de  $\eta_i^A$  et des départs totaux mesurés sur  $j$ ,  $\eta_j^D$ . Le nœud  $DN$  envoie ensuite quatre informations à son agrégateur de direction qui gère la direction entrante à laquelle  $DN$  appartient :  $\eta_j$ ,  $\lambda_i$ ,  $\mu_j$  et le temps actuel  $t$ .

L'agrégateur de direction maintient un timer  $\Delta_{DN} \geq \min\{\Delta_{SN}, \delta_{SN}\}$ . Lorsque ce compteur est atteint, il transmet les dernières données agrégées au nœud de décision.

## 5. 2. 2 Prédiction sur le nœud décision

Le second niveau de prédiction est sur le nœud de décision qui estime une valeur récente du nombre de véhicules sur chaque voie, sur la base de la dernière information dont il dispose. Le nœud de décision ajuste d'abord le nombre de véhicules qu'il possède en estimant les nouvelles arrivées sur la voie de chaque nœud  $DN\ j$ . Si l'on désigne par  $\eta_j(t)$  l'estimation du nombre de véhicules sur la voie commune entre le nœud  $SN\ i$  et le nœud  $DN\ j$  à la date actuelle  $t$ , et par  $\eta_j(\tau_j^{DN})$  la dernière comptabilisation d'arrivées reçue depuis le nœud  $DN\ j$  à la date  $\tau_j^{DN}$ , nous obtenons :

$$\eta_j(t) = \eta_j(\tau_j^{DN}) + (t - \tau_j^{DN} - \tau^s) \cdot \lambda_i .$$

Le temps de démarrage,  $\tau^s$ , ne dispose que d'une influence pour des fréquences très élevées de transmission. Lorsque le feu pour les mouvements concernés a été vert depuis la dernière mesure (et seulement dans ce cas), nous estimons les départs de façon similaire :

$$\eta_j(t) = \max(\eta_j(\tau_j^{DN}) - (t - \tau_j^{DN} - \tau^s) \cdot \mu_j, 0) .$$

## 5. 3 Gestion de plusieurs intersections

Nous choisissons de limiter la transmission des messages de synchronisation pour le seul cas où l'intersection de réception est moins chargée (et donc a un nombre total de véhicules plus faible) que l'intersection d'émission (sec. 4. 4). Un message arrive de manière asynchrone à une intersection qui pourrait avoir à faire face à plusieurs demandes au cours de la même phase. Dans ce cas, nous sélectionnons la stratégie qui

maximise le nombre attendu de véhicules qui traverseraient l'intersection au cours de la phase suivante, en ajoutant une contrainte sur les mouvements. Cette stratégie permet de réduire les probabilités d'encombrement et de perte de paquets sur le canal sans fil, en plus de ne pas transmettre des informations inutilement. Si des requêtes venant d'intersections voisines sont ignorées ou perdues, cela n'a pas grand impact, car les véhicules ne sont pas bloqués et continuent d'arriver et de partir : l'intersection peut les gérer localement. De plus, les messages peuvent être facilement retransmis.

Les opérations et les contraintes sur des intersections adjacentes sont les mêmes que celles décrites en début de chapitre. Cependant, ils sont amplifiés. En effet, les intersections adjacentes sont généralement suffisamment éloignées (chapitre 3), dans ce cas il n'y a plus d'interférences avec le réseau de capteurs, et par conséquent de perte de paquets. TAPIOCA laisse chaque intersection prendre sa propre décision localement – comme si elle était isolée – et essaye de synchroniser les intersections adjacentes avec des messages de synchronisation, la perte de paquets ne compromet donc pas le réseau. Elle provoque, dans le pire des cas, une décision locale au lieu d'une tentative pour créer une vague verte. De plus, les messages de synchronisation sont déjà affectés par notre mécanisme de prédiction, et sont datés : l'intersection réceptrice connaît donc bien le nombre de véhicules qui arrivent à un instant donné.

## 5. 4 Inventaire des messages

La figure 5.1 résume l'algorithme de TAPIOCA et le tableau 5.1 résume toutes les communications qui se produisent entre les capteurs. Les valeurs de la fréquence d'émission et de la taille des paquets sont des valeurs moyennes qui reflètent les simulations effectuées plus loin dans ce chapitre. Cependant, ces valeurs peuvent être adaptées à volonté pour un scénario particulier. Nous décrivons ci-après chaque message existant.

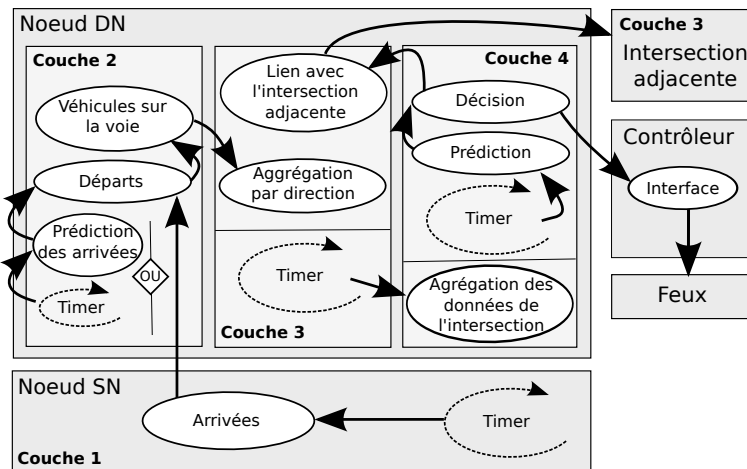


FIGURE 5.1 – Représentation des flux de communication et des principales procédures de notre algorithme.

| Messages                                     | Nœuds source et destination          | Fréquence                                                     | Taille     |
|----------------------------------------------|--------------------------------------|---------------------------------------------------------------|------------|
| Arrivées <sup>(a)</sup>                      | SN → DN                              | Tous les $\Delta_{SN}$                                        | 56 octets  |
| Comptabilisation des véhicules               | DN → agreg.                          | Tous les $\delta_{SN}$<br>ou lorsque <sup>(a)</sup><br>arrive | 184 octets |
| Agrégation<br>Comptabilisation des véhicules | agreg. → decision                    | Tous les $\Delta_{DN}$                                        | 472 octets |
| Caractéristiques de la phase<br>suivante     | decision → {contrôleur; agrégateurs} | 1 pkt/phase                                                   | 64 octets  |
| Synchronisation                              | agreg. →<br>agreg. voisin            | 1 paquet/phase<br>(asynchrone)                                | 200 octets |
| Signature des véhicules<br>(optionel)        | DN → DN voisin                       | 1 paquet/phase<br>(asynchrone)                                | 108 octets |

TABLE 5.1 – Résumé des communications.

Le premier type de messages contient les arrivées comptabilisées par un nœud *SN*. Il est envoyé à chaque déclenchement du timer  $\Delta_{SN}$  depuis un nœud *SN* au nœud *DN* situé en amont sur sa voie. Ce message étant transmis régulièrement, sa perte impliquerait juste une moins grande fraîcheur des données sur le nœud décision, et donc un plus grand taux d'erreur dans la perception du nombre de véhicules sur l'intersection.

Chaque nœud *DN* prend en compte le nombre de départs qu'il a comptabilisé afin de pouvoir calculer, à l'aide des arrivées reçues, le nombre de véhicules sur sa voie. Cette donnée est envoyée de façon régulière au nœud agrégateur responsable de sa direction. Ce message est transmis à chaque réception du message précédent ou à l'expiration du timer  $\delta_{SN}$ , provoquant une interpolation des arrivées. En cas de perte de ce message, de la même manière que précédemment, cela impliquerait juste une moins grande fraîcheur des données sur le nœud décision, étant donné que le message est retransmis régulièrement.

Chaque nœud agrégateur envoie, de manière régulière, un message agrégé contenant notamment les comptabilisations des véhicules des voies auxquelles il est rattaché. Ce message est à destination du nœud décision et est envoyé de façon régulière, toutes les  $\Delta_{DN}$  secondes. Sa perte provoque les mêmes effets que les deux cas précédents. Ce message faisant partie du processus de remontées et de comptabilisation du nombre de véhicules, un retard ou une perte n'impliquerait qu'un plus grand taux d'erreur au moment de décider de la prochaine phase, étant donné le délai plus important.

Un nœud décision est responsable du calcul des phases et envoie un message au contrôleur et aux nœuds agrégateurs. Le premier sert à lancer le feu vert. Sa perte implique donc de graves conséquences. Nous pouvons toutefois imaginer qu'un mécanisme de sécurité est présent à l'intérieur des contrôleurs, permettant d'éviter qu'une phase ne continue indéfiniment de se dérouler si aucun message n'est reçu. De même, nous pouvons facilement imaginer un mécanisme d'accusé de réception, qui ne serait pas coûteux étant donné que ce type de message n'est envoyé qu'une fois à chaque phase. Le deuxième message est envoyé aux nœuds agrégateurs afin que chacun puisse communiquer les informations relatives à la nouvelle phase sur les intersections voisines. Si ce message, ou le message envoyé par les nœuds agrégateurs aux intersections voisines, venait à être perdu, l'impact serait uniquement visible sur les performances

de notre algorithme. Les décisions se reposant avant tout sur un mécanisme de décision local, cela empêcherait – ou retarderait – probablement la création de vagues vertes.

## 5. 5 Simulations

### 5. 5. 1 Description des scénarios

Nous évaluons TAPIOCA en utilisant une co-simulation entre SUMO 0.17 et OMNeT++ 4.2 [VH08], liés ensemble à l’aide du framework Veins 2.1<sup>1</sup> [SGD11]. Notons que Veins est à la base conçu pour simuler un réseau véhiculaire. Toutefois, nous l’avons modifié de manière à intégrer un réseau de capteurs sans fil répondant aux caractéristiques qui nous intéressent (chapitre 3), et simuler des effets réseaux proches de la réalité (p. ex., perte de paquets). Les nœuds sont ainsi déployés selon l’architecture décrite sur la figure 3.1 et reposent sur le standard de communication IEEE 802.15.4 (*non-beacon enabled*), utilisant le protocole CSMA. Leur contrôleur d’interface réseau met en œuvre une carte d’interface réseau *Texas Instruments CC 2420 802.15.4*. Enfin, le modèle de propagation utilisé est le même que celui décrit à la section 3. 3. 3.

Les figures 5.2, 5.3, et 5.4 présentent des évaluations réalisées sur une et dix intersections adjacentes, générées aléatoirement par SUMO avec une intensité de trafic faible ( $\lambda = \frac{1}{3}$  véhicules par seconde) et une intensité de trafic supérieure (1 véhicule par seconde). Chaque simulation s’étale sur une durée de 3 600 s.

Nos résultats montrent la comparaison entre cinq méthodes. Tout d’abord, un plan de feux statique généré par SUMO. Ensuite, un algorithme adaptatif issu de la littérature ([YAKS10]) et TAPIOCA : les deux sont centralisés et ne sont pas soumis à des contraintes réseaux, ils fonctionnent indépendamment de l’équipement. Les deux dernières méthodes mettent en œuvre TAPIOCA réparti sur un réseau de capteurs sans fil respectivement avec et sans le mécanisme de prédiction décrit en 5. 2. Ces deux dernières méthodes sont simulées à l’aide de différentes valeurs de timers  $\Delta_{SN}$  et  $\Delta_{DN}$  : 1, 2, 4, 8, 16, 32 et 64 s, soit 49 configurations possibles.

Les résultats complets et d’autres scénarios (par exemple, des arrivées aléatoires ou en croissance) sont disponibles en ligne<sup>2</sup>.

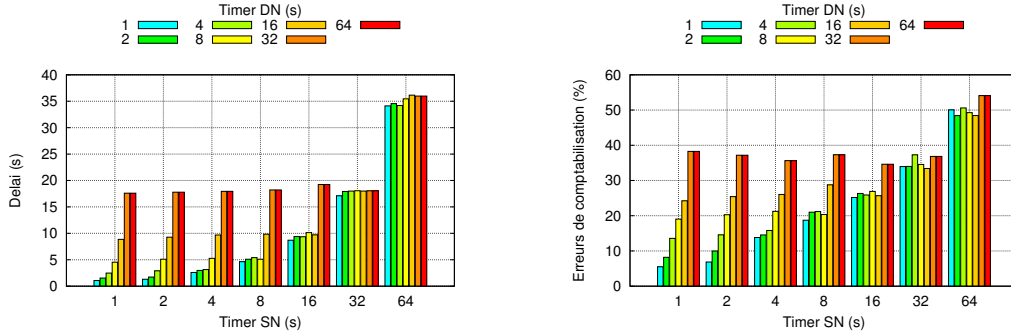
### 5. 5. 2 Evaluation des problèmes réseaux

Les figures en 5.2 nous montrent les problèmes qui sont rencontrés sur le réseau. La figure 5.2(a) représente le délai moyen entre les mesures et leur utilisation au niveau du nœud de décision. La figure 5.2(b) représente les erreurs de comptabilisation de véhicules. Ces résultats montrent que ces valeurs peuvent augmenter rapidement en fonction des valeurs de timers configurées. En configurant par exemple  $\Delta_{SN}$  et

1. <http://veins.car2x.org>

2. <http://tapioca.sfaye.com/WSN/>

$\Delta_{DN}$  à 16 secondes, nous constatons que le délai moyen entre la transmission d'un message et son utilisation est de 10 secondes. Ce délai, sur une simple intersection isolée, entraîne en moyenne 25% d'erreurs sur l'estimation du nombre de véhicules comptabilisés.



(a) Délai moyen entre la détection d'un véhicule et sa prise en compte par le nœud de décision.

(b) Erreur de comptabilisation moyenne.

FIGURE 5.2 – Simulations sur une intersection isolée (1/2).

### 5. 5. 3 Effet du mécanisme de prédiction

#### 5. 5. 3. 1 Performances sur une intersection isolée

Selon la figure 5.3(c) et en moyenne sur l'ensemble de nos simulations, nous constatons que  $\Delta_{DN}$  doit être au moins égal au quart de  $\Delta_{SN}$  pour s'assurer que l'estimation ne devienne pas exagérément fausse. Les figures 5.3(a) et 5.3(b) montrent l'efficacité de notre mécanisme de prédiction avec les deux flux d'arrivées décrits en introduction à cette section. L'axe des abscisses représente différentes combinaisons de timers  $\Delta_{SN}$  et de timers  $\Delta_{DN}$ .

Nous constatons en premier lieu que le timer  $\Delta_{DN}$  semble avoir un impact plus significatif que le timer  $\Delta_{SN}$ . En effet, le timer  $\Delta_{DN}$  est celui qui permet l'envoi des données au nœud de décision. Il est donc primordial et influe plus facilement sur les retards : quelle que soit la valeur du timer  $\Delta_{SN}$ , si le timer  $\Delta_{DN}$  est important, un délai sera toujours présent. À l'inverse, si  $\Delta_{DN}$  est faible et que  $\Delta_{SN}$  est élevé, les arrivées sont envoyées à moins grande fréquence, mais sont reçues très rapidement sur le nœud de décision. De plus, une phase peut couvrir cette fréquence d'envoi des arrivées, l'impact est donc plus faible.

Nous constatons également que la charge du système a un impact direct sur le comportement de notre algorithme, et donc sur le temps moyen d'attente des usagers. Pour une charge faible ( $\lambda = 1/3$ ), les temps moyens d'attente sont légers et l'amélioration suggérée par notre mécanisme d'interpolation est légère, mis à part pour des timers élevés. Pour une charge plus importante ( $\lambda = 1$ ), les temps moyens d'attente sont nettement plus élevés et les différences entre un scénario avec et sans mécanisme d'interpolation sont importantes. Nous constatons par ailleurs que notre mécanisme

d'interpolation semble plus efficace lorsque le trafic est plus important, ceci pouvant être expliqué par une plus grande régularité dans les arrivées de nouveaux véhicules.

### 5. 5. 3. 2 Performances sur plusieurs intersections

Pour ces évaluations, nous avons choisi de présenter un scénario intermédiaire, en réglant  $\Delta_{SN} = \Delta_{DN} = 8$  secondes.

Les résultats de simulation présentés sur les figures 5.4(a) et 5.4(b) montrent que toutes les implémentations de TAPIOCA réalisent de meilleurs temps d'attente et de trajets moyens que ceux générés par des plans de feux prédéterminé de SUMO : 93,8 % meilleurs pour la version centralisée, 88 % pour la version réseau de capteurs sans fil avec notre mécanisme de prédiction et 75 % meilleur pour la version de base avec un réseau de capteurs sans fil uniquement.

La figure 5.4(c) montre la proportion de scénarios dans lesquels chaque méthode est la plus efficace, en fonction du nombre de véhicules qui atteignent le meilleur temps d'attente. Nous pouvons voir que TAPIOCA est presque toujours le meilleur choix, et que la mise en œuvre avec un réseau de capteurs sans fil et la méthode de prédiction est la meilleure solution dans 40 % des cas.

Certains résultats semblent nous indiquer que TAPIOCA, avec un réseau de capteurs sans fil et le mécanisme de prédiction, est plus efficace que la version centralisée (par exemple, sur les figures 5.3(b) et 5.4(c)). Nous pouvons supposer que notre méthode de prédiction, en plus de prendre en considération les retards de transmission, permet d'intégrer – à l'aide du taux d'arrivée – des véhicules que les capteurs n'ont eux-mêmes pas encore détectés.

## 5. 6 Conclusion

Dans ce chapitre, nous proposons et évaluons un algorithme adaptatif de contrôle des feux de circulation, distribué sur plusieurs intersections et qui utilise un réseau de capteurs sans fil. Nous montrons l'efficacité du réseau de capteurs sans fil et l'intérêt de gérer le trafic localement sur une intersection. Nous proposons notamment un mécanisme de prévision qui nous permet d'alléger les effets provoqués par les retards et les pertes de paquets.

Bien entendu, des pistes restent à aborder. Par exemple, nous pouvons imaginer étudier l'économie d'énergie et limiter les communications, en changeant dynamiquement les timers sur la base de la stabilité du taux d'arrivée et du taux de départ. De plus, nous limitons ici la communication entre les intersections directement voisines. Cependant, nous pourrions bénéficier d'informations provenant d'une plus grande distance, en particulier lors de la construction de vagues vertes, même si nous avons le *sensiment* que cette opération serait plus coûteuse que bénéfique.

Dans les évaluations réalisées en section 5. 5, nous nous sommes reposés sur un faible nombre de scénarios. Le paramètre  $\alpha$  du filtre EWMA a été réglé de façon empirique afin que l'estimation des véhicules soit la plus juste possible. Cela nous a

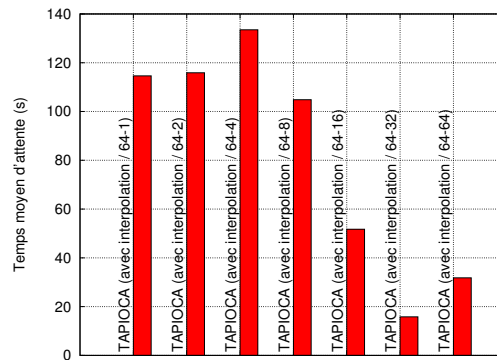
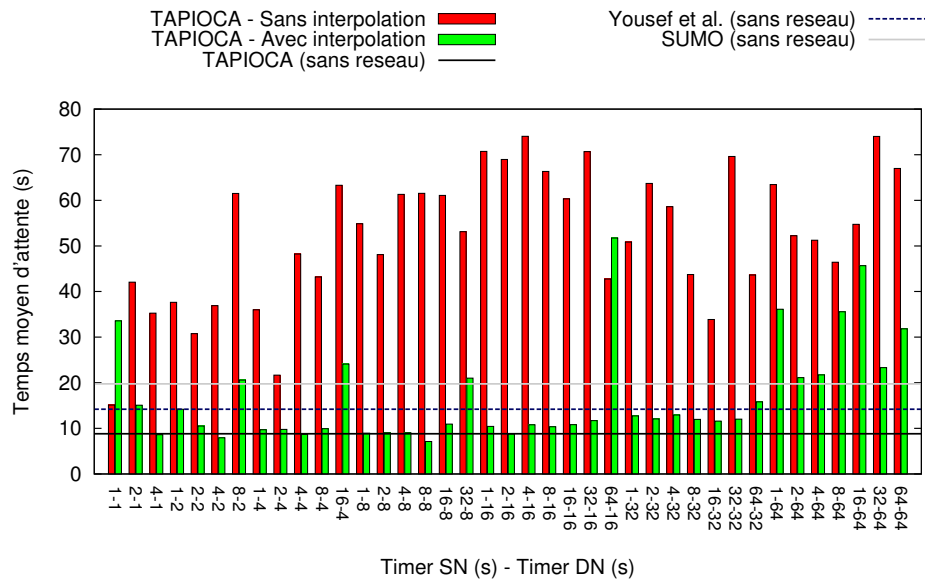
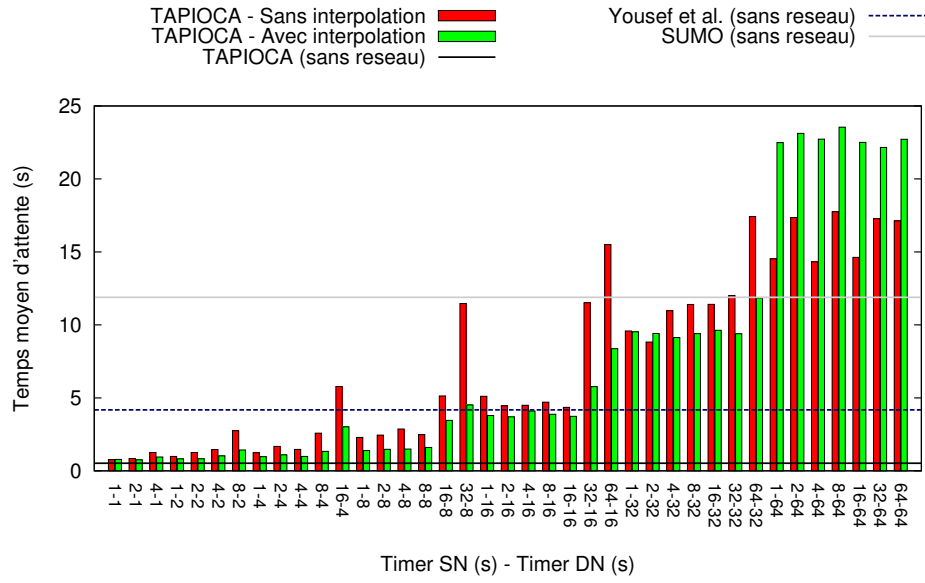
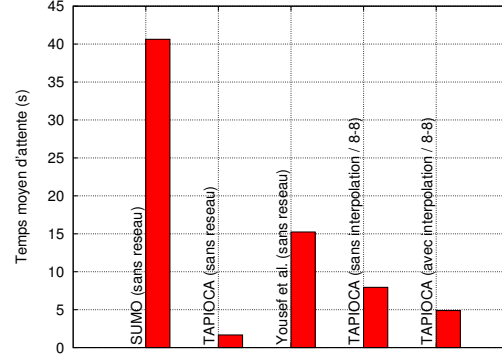
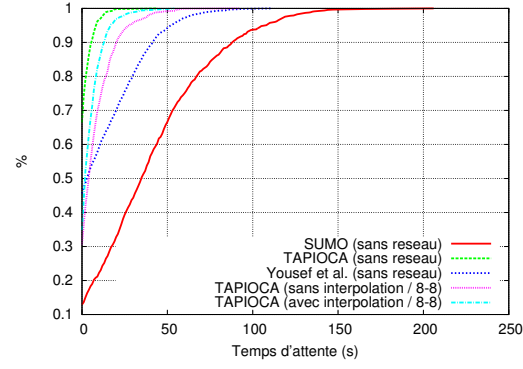


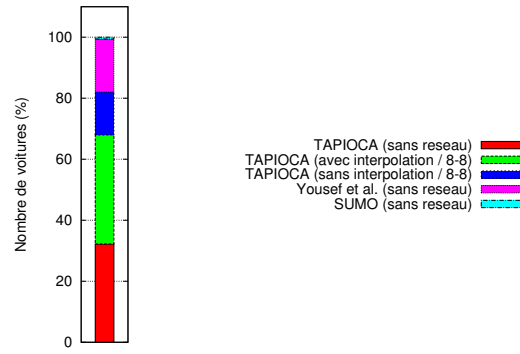
FIGURE 5.3 – Simulations sur une intersection isolée (2/2).



(a) Temps moyen d'attente.



(b) Temps moyen d'attente (CDF).



(c) Répartition de la meilleure stratégie.

FIGURE 5.4 – Simulations sur dix intersections adjacentes avec  $\lambda = \frac{1}{3}$  et  $\Delta_{SN} = \Delta_{DN} = 8$ .

laissé pensé que dans un cas général, ce paramètre variait en fonction de la stabilité du trafic. Selon nos observations, plus le trafic est stable (ou important), plus le paramètre  $\alpha$  peut être élevé. À l'inverse, plus il est instable (ou faible), et plus le paramètre  $\alpha$  peut être bas, afin de réagir plus rapidement aux variations de trafic. Une perspective de ce travail seraient d'étudier ces aspects, et notamment ce deuxième cas, à mesure qu'un paramètre  $\alpha$  trop faible n'irait peut-être pas avec les contraintes réseaux (notamment les pertes de paquets).

Les résultats présentés dans ce chapitre nous laissent penser que TAPIOCA possède davantage de difficultés quand la charge est trop importante. Si le mécanisme d'interpolation suggéré semble bien se comporter dans nos simulations, il nous faut toutefois étudier l'impact que peuvent avoir des scénarios avec un trafic fortement encombré. Dans le chapitre suivant, nous proposons donc d'étudier les cas d'embouteillages, afin de comprendre les différences fondamentales avec des scénarios plus classiques.

