Un algorithme TDMA réparti pour les réseaux de capteurs

Ted Herman — Sébastien Tixeuil

University of Iowa, Department of Computer Science
LRI - CNRS UMR 8623, INRIA Projet Grand Large, Université Paris-Sud XI

Les méthodes TDMA utilisent le multiplexage temporel et planifient les transmissions pour éviter les collisions de trames. Cependant, ces méthodes souffrent d'un problème de dimensionnement quand elles sont utilisées dans des réseaux *ad hoc* à topologie dynamique et sujets à des défaillances de nœuds. Cet article propose un algorithme réparti pour la planification TDMA qui résiste de manière auto-stabilisante aux défaillances transitoires et aux changements dynamiques de topologie. Le temps de convergence est $O(1)$ indépendamment de la taille du réseau si celui-ci admet une borne constante sur la taille du voisinage de chaque nœud. Le détail de l'algorithme et les preuves peuvent être trouvées dans [5].

1 Introduction

L'évitement et la gestion des collisions sont des aspects fondamentaux dans les protocoles pour les réseaux sans fils. Les réseaux actuellement envisagés pour les capteurs [12] et les petits dispositifs [3] requièrent des caractéristiques comme les économies d'énergie, la capacité à passer à l'échelle, la tolérance aux défaillances transitoires, et l'adaptativité aux changements de topologie. Indirectement, un protocole de communication qui permet d'éviter les collisions permet d'économiser l'énergie, puisque le besoin de retransmettre un message s'en trouve réduit. L'accès au medium de communication par multiplexage temporel (TDMA) est une technique raisonnable pour éviter les collisions, cependant les priorités liées au passage à l'échelle et à la tolérance aux pannes n'ont pas été mises en avant par la plupart des travaux précédents. Le problème algorithmique de l'allocation des créneaux de temps dans TDMA est lié au problème classique de l'allocation des fréquences dans FDMA. Ces problèmes peuvent être formulés comme des problèmes de coloration contraint des sommets d'un graphe [9]. Pour FDMA, chaque couleur représente une fréquence, et pour éviter les collisions, on assure que tous les sommets à distance deux (ou moins) les uns des autres ont des couleurs différentes. Une contrainte supplémentaire est que les couleurs choisies par des sommets voisins sont suffisamment distantes pour éviter les interférences. Si l'ensemble des couleurs utilisées est l'intervalle des entiers $[0,\lambda]$, alors les couleurs (f_v, f_w) des sommets voisins (v,w) doivent satisfaire $|f_v - f_w| > 1$ pour éviter les interférences. La notation standard pour exprimer une telle contrainte est $L(\ell_1,\ell_2)$: pour toute paire de sommets à distance $i \in \{1,2\}$, les couleurs diffèrent d'au moins ℓ_i. La coloration d'un graphe pour FDMA devrait donc satisfaire la contrainte $L(2,1)$. De plus, une solution qui optimise le nombre de couleurs utilisées est préférable, puisqu'elle réduit le nombre de fréquences nécessaires.

Le problème de la coloration dans TDMA est légèrement différent. Soit $L'(\ell_1,\ell_2)$ la contrainte que pour toute paire de sommets à distance $i \in \{1,2\}$, les couleurs diffèrent de ℓ_i mod $(\lambda+1)$. Cette contrainte exprime les problèmes aux bordures des créneaux de temps (voir également [8]). La contrainte de coloration usuelle pour TDMA est $L'(1,1)$. Si les créneaux sont imprécis (par exemple parce que la synchronisation dans le temps n'est pas parfaite), il est possible de demander une séparation plus stricte des couleurs, comme $L'(2,2)$. Dans cet article, nous nous restreignons à la contrainte $L'(1,1)$, équivalente à $L(1,1)$, mais notre approche peut être étendue pour satisfaire $L'(2,2)$. Minimiser le nombre de couleurs pour TDMA

est souhaitable, car si une période de temps correspondant à la séquence de couleurs $0..\lambda$ est ramenée à l'intervalle unitaire $[0,1]$, chaque couleur représente une fraction $1/(\lambda+1)$ de la bande passante. Donc plus λ est petit et mieux la bande passante est utilisée. Les problèmes de coloration avec les contraintes $L(1,0)$, $L(0,1)$, $L(1,1)$, et $L(2,1)$ ont été largement étudiés non seulement dans les graphes généraux mais aussi dans de nombreux graphes spéciaux [2, 6, 10]. La plupart de ces problèmes sont NP-complets et bien que des algorithmes d'approximation aient été proposés, de tels algorithmes ne sont typiquement pas répartis.

Notre Contribution. Nos travaux portent sur les algorithmes d'allocation de créneaux de temps dans un réseau qui évolue dynamiquement, sujet à des défaillances transitoires, en conservant de bonnes propriétés lors du passage à l'échelle. Notre approche pour gérer à la fois la dynamicité du réseau et les défaillances transitoires est celle de l'auto-stabilisation, qui assure que le système converge vers une allocation TDMA valide après une défaillance transitoire ou un changement de topologie. Nous assurons le passage à l'échelle en proposant un algorithme probabiliste d'allocation de créneaux de temps dont le temps moyen de convergence est $O(1)$. La base de notre algorithme consiste en une technique de regroupement rapide probabiliste, qui pourrait être exploitée afin de résoudre d'autres problèmes dans les réseaux de capteurs.

Travaux précédents. Kulkarni et Arumugam [7] ont été les premiers à développer l'auto-stabilisation de TDMA pour les réseaux de capteurs. Ils partent d'une topologie en grille (extensible à toute topologie par plongement dans la grille) et supposent que chaque sommet connaît sa position dans la grille (cette position est utilisée pour calculer l'allocation des créneaux). Utiliser leur approche dans des graphes généraux requiert que le plongement dans la grille soit le même pour tous les sommets et connu avant la mise en œuvre de l'algorithme. Par conséquent, et à la différence de notre approche, leur algorithme ne peut être utilisé dans les réseaux dynamiques. Notre solution emprunte des techniques de coloration auto-stabilisantes [4]. La nouveauté ici tient au fait que le modèle de communication est seulement probabilistiquement correct (en raison de la possibilité des collisions lors de l'accès au medium). Par ailleurs, notre coloration utilise un nombre constant de couleurs pour le problème $L(1,1)$, alors que la solution précédemment connue utilise n^2 couleurs.

2 Réseaux de capteurs sans fil

Le système est constitué d'un ensemble V de nœuds dans un réseau sans fil *ad hoc*. Les communications entre les nœuds utilisent une radio à faible puissance. Chaque nœud p peut communiquer avec un sous-ensemble $N_p \subseteq V$ (appelé voisinage de p) de nœuds déterminé par la portée de son signal radio. Dans le modèle sans fil, une transmission est omnidirectionelle : chaque message envoyé par p est en fait diffusé à tous les nœuds de N_p. Nous supposons que les communications sont bidirectionnelles. Nous appelons N_p^i le voisinage à distance i de p. La distribution des nœuds est telle que le réseau est connexe. En outre, la distribution est suffisamment dispersée (du point de vue de la portée radio) pour que l'on puisse borner la taille du voisinage de tout nœud : il existe Δ tel que pour tout p, $|N_p| \leq \Delta$.

Chaque nœud dispose d'une horloge temps-réel. Nous supposons que tous les nœuds sont synchronisés par une même horloge globale. Pour simplifier la présentation, nous supposons que chaque nœud possède un identifiant unique. Chaque nœud utilise la même fréquence radio (une même fréquence est partagée spatialement par tous les nœuds du réseau). Quand p transmet un message, il est incapable de détecter si un autre nœud est également en train de transmettre, or des collisions peuvent survenir. Les nœuds ne possèdent aucun dispositif matériel de détection des collisions. Si $q \in N_p$ et $r \in N_p$ transmettent simultanément, alors p reçoît la superposition de leurs transmissions, mais p ne peut détecter que la superposition est le résultat d'une collision, car il peut également s'agir d'une corruption due au bruit. Nous supposons que chaque message comprend suffisamment de code de détection d'erreurs pour que le résultat d'une corruption ou d'une collision puisse être détecté – un nœud ne peut cependant pas distinguer une corruption d'une collision.

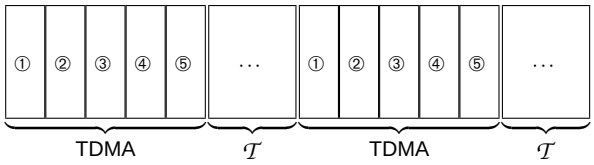
Ces contraintes empêchent l'utilisation de CSMA/CD pour l'accès au médium radio. Toutefois, certaines techniques de base de CSMA/CA restent applicables : si le nœud p a un message à transmettre, mais qu'il reçoît un signal, alors p ne commence à transmettre que lorsqu'il détecte une absence de signal. Une technique statistique pour gérer les collisions dans ce modèle est CSMA/CA : avant de transmettre un message,

p attend un temps aléatoire. Nous supposons que les nœuds possèdent un dispositif CSMA/CA (comme implanté, par exemple, dans [11]). Nous supposons que l'implantation CSMA/CA satisfait la propriété : il existe une constante $\tau > 0$ telle que la probabilité d'une transmission sans collision est au moins τ (ceci correspond à des hypothèses habituelles dans ce cadre, cf. [1]). L'indépendance de τ pour la transmission de plusieurs trames indique que nous supposons une distribution probabiliste sans mémoire.

3 Construction d'un modèle sans collision

Notre but est d'implanter un service de communication général sans collision. Ce service peut être vu comme une transformation du modèle donné section 2 dans un modèle sans collision. Un tel service simplifie l'écriture des applications et peut réduire l'énergie nécessaire aux communications (les messages n'ont plus besoin d'être retransmis). Soit $\mathcal{T}$ la tâche consistant à convertir le modèle de la section 2 dans un modèle sans collision. Nous souhaitons obtenir une solution pour $\mathcal{T}$ qui soit auto-stabilisante. Après que des défaillances transitoires ou des reconfiguration se sont produites, les nœuds peuvent se trouver dans un état inconsistant où des collisions peuvent survenir. Le transformateur $\mathcal{T}$ doit corriger les états des nœuds pour revenir automatiquement à des communications sans collision. Bien que notre but soit de proposer aux applications un service sans collision, notre implantation peut introduire des messages supplémentaires sujets aux collisions. Initialement, nous acceptons la collision puis retransmission de ces messages, qui sont internes à $\mathcal{T}$ et invisibles de l'application.

Pour résoudre $\mathcal{T}$, il suffit d'affecter à chaque nœud une couleur, et d'utiliser ces couleurs pour allouer des créneaux de temps [9]. Avant l'allocation des créneaux, nous divisons le temps radio en deux parties : une partie est réservée au multiplexage TDMA des messages de l'application ; l'autre partie est réservée à l'algorithme de coloration et d'allocation des créneaux. Le diagamme suivante présente une partie TDMA à 5 créneaux, et une partie fixe pour le transformateur $\mathcal{T}$. La résolution de $\mathcal{T}$ se déroule dans la concaténation de tous les créneaux prévus à cet effet (les créneaux $\mathcal{T}$). Alors que CSMA/CA est utilisé pour gérer les collisions dans les créneaux $\mathcal{T}$, les créneaux TDMA restants n'utilisent pas de délai aléatoire. Au cours de l'initialisation ou après un changement dynamique de topologie, des trames peuvent entrer en collision dans les créneaux TDMA, mais après que l'algorithme d'allocation a stabilisé, les collisions ne peuvent plus se produire dans les créneaux TDMA.



Le reste de la description du transformateur pour $\mathcal{T}$ consiste à développer une séquence d'algorithmes qui facilite son implantation. Ces algorithmes se doivent d'être auto-stabilisants, et sont : l'identification du voisinage, le nommage unique de voisinage, la construction d'un ensemble maximal indépendant, la coloration minimale de voisinage, et l'allocation des créneaux de temps.

Identification du voisinage. Etant donné que nous considérons des réseaux de type *ad hoc*, et que la topologie de ces réseaux est dynamique, nous n'avons pas supposé que les nœuds possédaient des informations sur leur voisinage. Un algorithme est donc nécessaire pour identifier ce voisinage. Nous décrivons maintenant comment un nœud p peut obtenir la connaissance de N_p^i, pour $i \in 1,2,3$. Chaque nœud peut avoir une représentation de N_p et de N_p^2 par une liste d'identifiants obtenue *via* les messages reçus par p. Cependant, comme nous ne faisons pas de suppositions quant à l'état initial d'un nœud, ces listes peuvent avoir initialement des données arbitraires. La technique de base de p, pour toute variable partagée L_p (c'est à dire toute variable dont la valeur est destinée à être communiquée au voisinage N_p), consiste à transmettre régulièrement L_p. De manière duale, p stocke la valeur des variables partagées par son voisinage, en y ajoutant l'heure de dernière mise à jour, de telle sorte que les variables partagées par le voisinage de p

n'ayant pas été mises à jour depuis longtemps sont supprimées. La construction du voisinage N_p se fait simplement en transmettant régulièrement l'identifiant unique de tous les nœuds de N_p, et les voisinages N_p^2 et N_p^3 peuvent être construits de manière similaire, les valeurs erronées étant supprimées quand elles ne sont plus rafraîchies. Cette méthode est auto-stabilisante seulement si les variables partagées ne changent pas de valeur pendant un temps supérieur au temps de stabilisation . À l'exception de l'algorithme de coloration de voisinage, tous nos algorithmes possèdent cette propriété.

Pour les algorithmes décrits ci-après, nous demandons plus qu'une cohérence au bout d'un certain temps. Nous demandons qu'avec une probabilité constante, toute variable partagée L_p est mise en cache dans N_p en temps constant. Ceci équivaut à demander qu'avec probabilité constante, un nœud transmet en temps constant et ce, sans collision. Pour ce faire, un nœud p espace les transmissions de ses variables partagées de $\kappa + \beta$, où β est une variable aléatoire. Cet espacement implique que pour pour instant t entre deux transmissions, il existe une probabilité constante pour que la prochaine transmission survienne avant $t + \alpha$ pour une constante α donnée. La probabilité combinée d'attendre au plus α puis d'envoyer sans collision est bornée inférieurement par une constante. Il s'en suit que le nombre moyen de tentatives avant de propager sans collision une variable partagée est en $O(1)$. Pour la suite, le temps moyen pour la propagation des variables est supposé constant.

Nommage unique de voisinage. Un algorithme qui effectue un nommage unique de voisinage donne à chaque nœud un nom distinct de celui de ses N^3 voisins. Ceci peut sembler bizarre étant donné que nous supposons que les nœuds possèdent des identifiants uniques, mais si on essaie d'utiliser ces identifiants pour de la coloration, l'ensemble potentiellement grand des identifiants peut poser problème lors du passage à l'échelle. Il est donc intéressant d'affecter des noms, tirés d'un espace de taille constante, en s'assurant qu'ils sont localement uniques. Ce problème peut être vu comme une coloration de N^3 (satisfaisant $L(1,1,1)$) et suggère une solution immédiate à $\mathcal{T}$ en affectant à chaque couleur un créneau de temps. Cependant, cette solution gaspillerait de la bande passante si l'espace des identifiants était plus grand que $|N^3|$. Nous utilisons donc ce nommage unique de voisinage pour construire par la suite une coloration de N^2 qui soit minimal. Notre algorithme de nommage unique est basé sur la technique probabiliste de [4], et introduit de nouvelles caractéristiques. Soit $\gamma = \lceil \Delta^t \rceil$ pour un $t > 3$[†]. Si un nœud ne possède pas une couleur unique (choisie entre 0 et γ) dans son cache de N_p^3, il choisit une nouvelle couleur aléatoirement parmi les couleurs disponibles. Une propriété clef de notre algorithme (que ne possède pas [4]) est la suivante : la propriété d'unicité de la couleur d'un nœud p est stable. De manière similaire, la propriété d'unicité de tout sous-ensemble des nœuds est également stable. En d'autres termes, une fois qu'un nœud est considéré comme unique pour tous les voisinages auxquels il appartient, il est stable. Nous pouvons par conséquent raisonner à partir d'un modèle Markovien sur les exécutions et montrer que la probabilité d'une séquence d'actions menant d'un ensemble stable donné à un ensemble stable plus grand est positive. Le nommage unique de voisinage que nous proposons possède une propriété que les identifiants globaux du système n'ont pas : comme les identifiants sont de taille constante, la plus longue chaîne d'identifiants croissants dans le graphe est également de taille constante. Nous supposons dorénavant que les identifiants des nœuds sont ceux obtenus par notre algorithme de nommage unique, et qu'ils possèdent la propriété pré-citée.

Ensemble maximal indépendant. De manière à passer d'identifiants chromatiques choisis parmi γ couleurs à une coloration de N^2 minimal (pas minimum, qui est un problème NP-difficile), nous utilisons une élection locale parmi les nœuds de manière à obtenir un ensemble maximal indépendant de nœuds provilégiés. Les identifiants obtenus induisent un graphe orienté acyclique (en orientant les arêtes du graphe du plus grand identifiant au plus petit). Les racines de ce graphe (*i.e.* les nœuds dont l'identifiant est localement minimal) sont privilégiées. Par la suite, un nœud devient privilégié si et seulement si aucun de ses prédécesseurs (dans le graphe orienté acyclique) n'est lui même privilégié. Il est facile de montrer que cet algorithme converge de manière auto-stabilisante vers un ensemble maximal indépendant de nœuds privilégiés en un

[†] Il y a un compromis à trouver pour le choix de t dans $\gamma = \Delta^t$. D'abord, t devrait être suffisamment grand pour que le choix d'un nouvel identifiant soit unique avec grande probabilité. Dans le pire des cas, $|N_p^3| = \Delta^3 - \Delta^2 + \Delta + 1$, et le cache de chaque nœud peut contenir $\Delta^3 - \Delta^2 + \Delta$ noms, donc le choix $t \approx 6$ peut s'avérer satisfaisant. En général, de grandes valeurs de t diminuent le temps de convergence moyen de l'algorithme de nommage unique de voisinage, et de petites valeurs de t réduisent la constante d, qui à son tour réduit le temps de convergence moyen des algorithmes décrits ci-après.

temps proportionnel à la hauteur du graphe orienté acyclique (donc en un temps constant puisque nous avons montré que ce graphe était de hauteur constante).

Coloration minimale de voisinage. Notre méthode de coloration à distance 2 est simple : les couleurs sont affectées par les nœuds privilégiés. Les conflits éventuels (deux nœuds privilégiés peuvent être à distance 2 ou 3 et demander l'affectation de couleurs contradictoires à un nœud faisant partie de leur voisinage commun) sont résolus en utilisant l'identifiant des nœuds privilégiés en conflit ; de ce qui précède, cet identifiant est forcément différent pour les nœuds situés à distance 3 ou moins. Comme les nœuds privilégiés affectent les couleurs en prenant à chaque fois la plus petite possible, et que cette affectation, pour un nœud privilégié donné, se fait essentiellement de manière séquentielle le long des chemins d'identifiants croissants, la coloration de N^2 est minimale, et converge en temps constant (la hauteur des conflits est au plus 3).

Attribution des créneaux de temps. Étant donné une coloration à distance 2 des nœuds du réseau, il reste à attribuer les créneaux de temps à chaque nœud pour TDMA. Nous supposons ici que tous les nœuds sont de priorité égale pour l'accès au médium. Ceci implique que chaque nœud devrait avoir autant de bande passante que les autres. Par exemple, considérons un nœud p qui a trois couleurs dans N_p^2, mais qui possède un voisin q avec 4 couleurs dans N_q^2. Le nœud q peut utiliser au plus $1/4$ de la bande passante, alors que p peut en utiliser au plus $1/3$. L'équité entre les nœuds n'est pas violée si on autorise p à utiliser $1/3$ de la bande passante si par ailleurs cette bande passante serait perdue. Notre algorithme d'allocation TDMA calcule donc pour chaque nœud le nombre de couleurs visibles dans son voisinage à distance 2, puis procède des nœuds les plus contraints (ceux qui voient le plus de couleurs) vers les nœuds les moins contraints. Le nombre de couleurs dans un voisinage à distance 2 étant borné par une constante (de l'ordre de Δ^2), le temps de convergence avant stabilisation est également constant.

Conclusion Notre algorithme permet une communication sans collision dans un réseau de capteurs sans fils. Il est auto-stabilisant et peut de ce fait tolérer des défaillances transitoires, mais peut également passer à l'échelle puisque son temps de stabilisation est $O(1)$. Certains composants de notre algorithme peuvent trouver des applications dans les réseaux *ad hoc*. Le nommage unique de voisinage et la construction de l'ensemble maximal indépendant peuvent être vus comme des regroupements locaux, construits en temps $O(1)$, qui peuvent être utilisés dans d'autres applications.

Références

[1] D. Bertsekas, R. Gallager. *Data Networks*, Prentice-Hall, 1987.

[2] H. L. Bodlaender, T. Kloks, R. B. Tan, J. van Leeuwen. Approximations for λ-coloring of graphs. University of Utrecht, Department of Computer Science, Technical Report 2000-25, 2000 (25 pages).

[3] D. E. Culler, J. Hill, P. Buonadonna, R. Szewczyk, A. Woo. A network-centric approach to embedded software for tiny devices. In *Proceedings of Embedded Software, First International Workshop EMSOFT 2001*, Springer LNCS 2211, pp. 114-130, 2001.

[4] M. Gradinariu, C. Johnen. Self-stabilizing neighborhood unique naming under unfair scheduler. In *Euro-Par'01 Parallel Processing, Proceedings*, Springer LNCS 2150, 2001, pp. 458-465.

[5] T. Herman, S. Tixeuil. A Distributed TDMA Slot Assignment Algorithm for Wireless Sensor Networks. *LRI Technical Report 1370*, Sep. 2003.

[6] S. O. Krumke, M. V. Marathe, S. S. Ravi. Models and approximation algorithms for channel assignment in radio networks. *Wireless Networks* 7(6 2001) :575-584.

[7] S. S. Kulkarni, U. Arumugam. Collision-free communication in sensor networks. In *Proceedings of Self-Stabilizing Systems, 6th International Symposium*, Springer LNCS 2704, 2003, pp. 17-31.

[8] D. Liu, On sizes of Graphs with Fixed Orders and Spans of Circular Distance Two Labellings, *Ars Comb.*, 67 (2003), 125 – 139.

[9] S. Ramanathan. A unified framework and algorithm for channel assignment in wireless networks. *Wireless Networks* 5(2 1999) :81-94.

[10] S. Ramanathan, E. L. Lloyd. Scheduling algorithms for multi-hop radio networks. *IEEE/ACM Transactions on Networking*, 1(2 1993) :166-177.

[11] A. Woo, D. Culler. A transmission control scheme for media access in sensor networks. In *Proceedings of the Seventh International Conference on Mobile Computing and Networking (Mobicom 2001)*, pp. 221-235, 2001.

[12] F. Zhao, L. Guibas (Editors). *Proceedings of Information Processing in Sensor Networks, Second International Workshop, IPSN 2003*, Springer LNCS 2634. April, 2003.