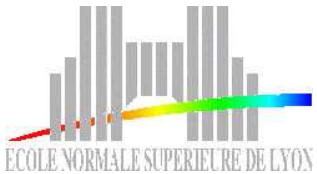
TCP et réseaux ad hoc
L'évitement de la congestion

Mathias Péron

stage de recherche MIM2 – septembre 2004



Introduction – les réseaux ad-hoc

Description

– réseaux locaux sans fil utilisant le médium radio
– chaque entité est mobile et autonome
– le réseau est auto-configurable : aucune station de base nécessaire, chaque nœud peut agir comme client et comme routeur

Applications réseaux spontanés, *sensor-networks*

Problématiques antennes intelligentes ▷ équité d'accès au médium ▷ algorithmes de routage ▷ protocoles de transport ▷ sécurité, QoS

Introduction – cadre de travail

– couche *Medium Access Control* (MAC) de l'IEEE 802.11b

– le protocole AODV au niveau réseau

Simulations sous *NetworkSimulator2* (NS2)

On va travailler dans les chaînes pour éviter les problèmes connus.

Plan de l'exposé

(i) le protocole TCP

(ii) TCP dans un environnement ad hoc

(iii) Mécanismes et solutions proposées

(iv) notre solution, TCPToK

TCP - propriétés

➔ protocole de la couche transport

➔ 80% du trafic Internet

Propriétés

– orientation à la connexion

– fiabilité

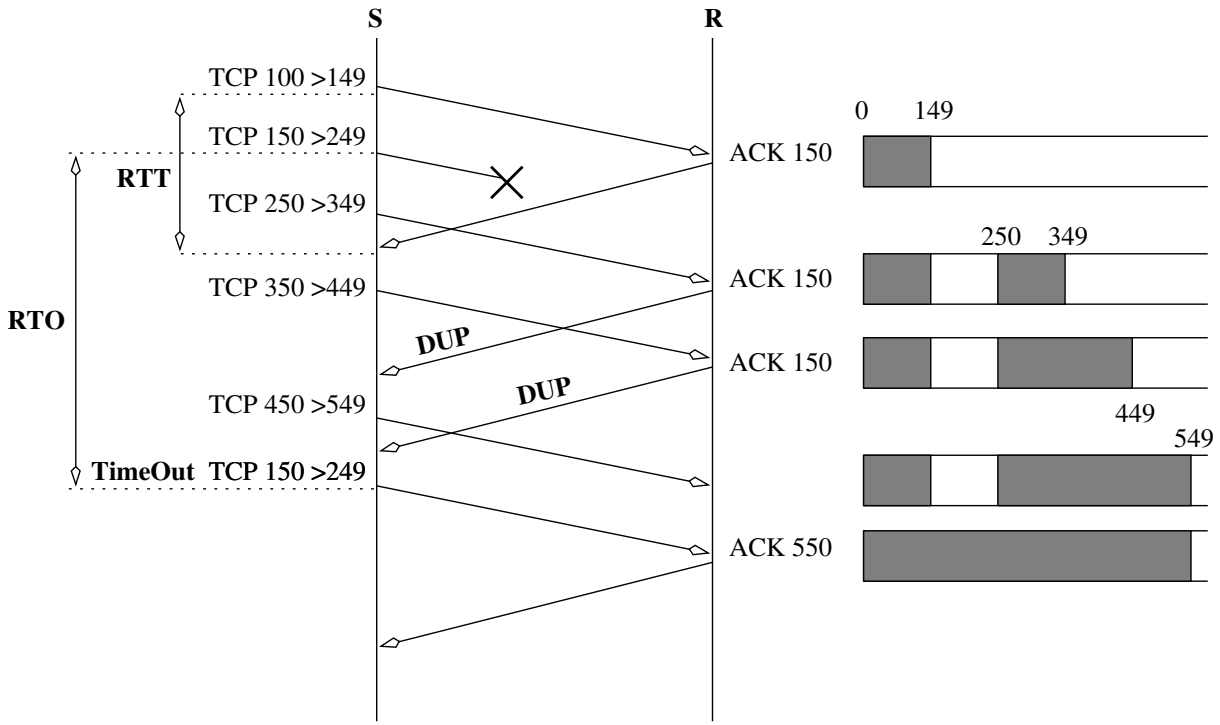
– sémantique *end-to-end* (bout à bout)

– indépendant vis à vis des données

– régulation du débit à travers un contrôle de flux et un contrôle de congestion

TCP - mécanismes pour la fiabilité

Acquittements (ACK) accumulatifs $\rightsquigarrow$ SACK

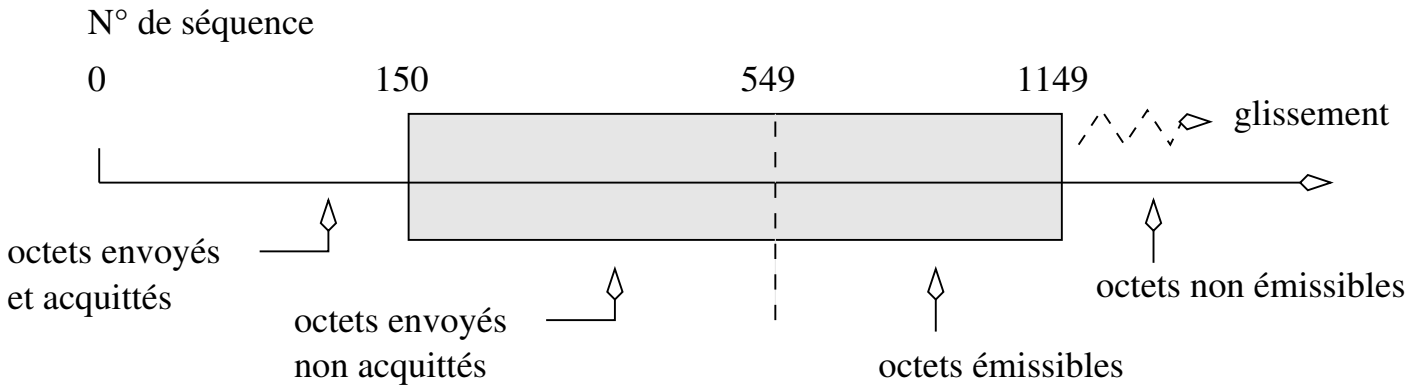


Retransmissions si *Retransmission TimeOut* (RTO) ou si x acquittements dupliqués

TCP - contrôle de la congestion (1)

Régulation du débit $\hookrightarrow$ mécanisme de fenêtre d'envoi

La fenêtre glisse au fur et à mesure que les ACKs arrivent.



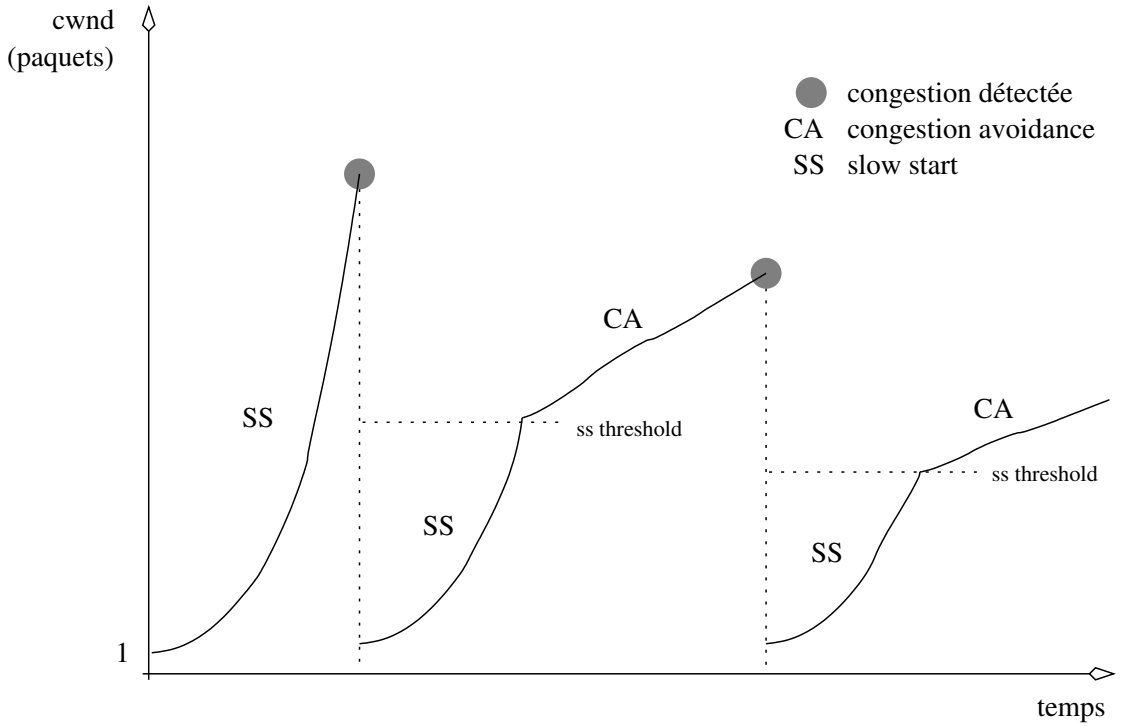
sliding windows

Retransmissions rapides $\rightsquigarrow$ Reno, NewReno

TCP - contrôle de la congestion (2)

Idée jauger la congestion du réseau

Indices : timeouts, ACKs dupliqués $\rightsquigarrow$ ECN

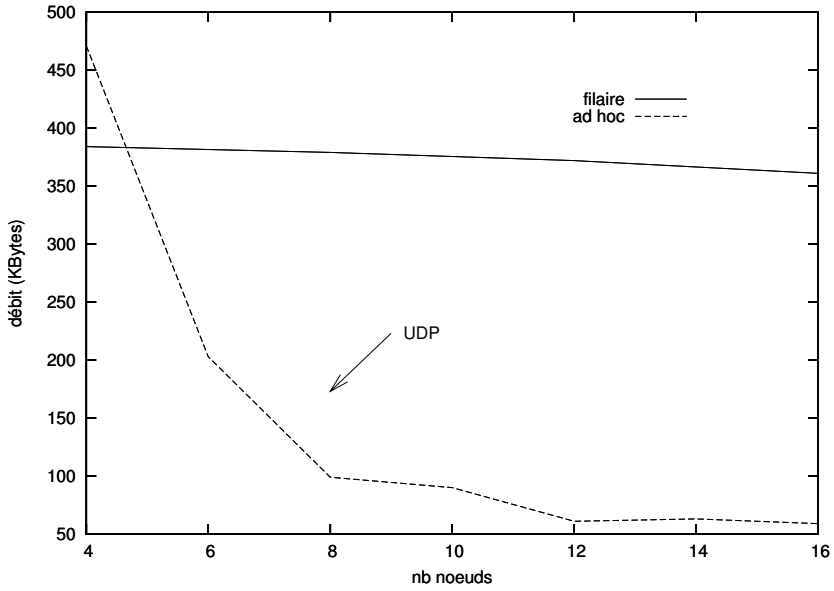


comportement de la fenêtre de congestion (CWND)

TCP ○ Ad hoc - comportement

Soucis baisse anormale du débit avec la longueur de la chaîne

Pertes de paquets de données : TCP 5 %, UDP 60 %

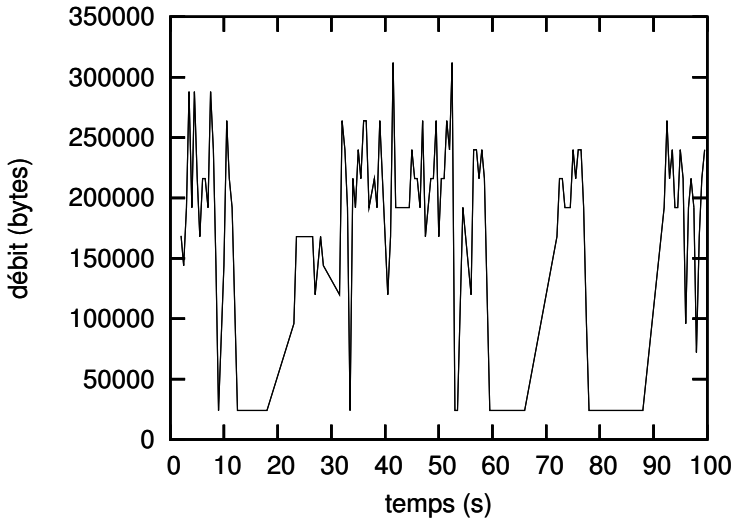


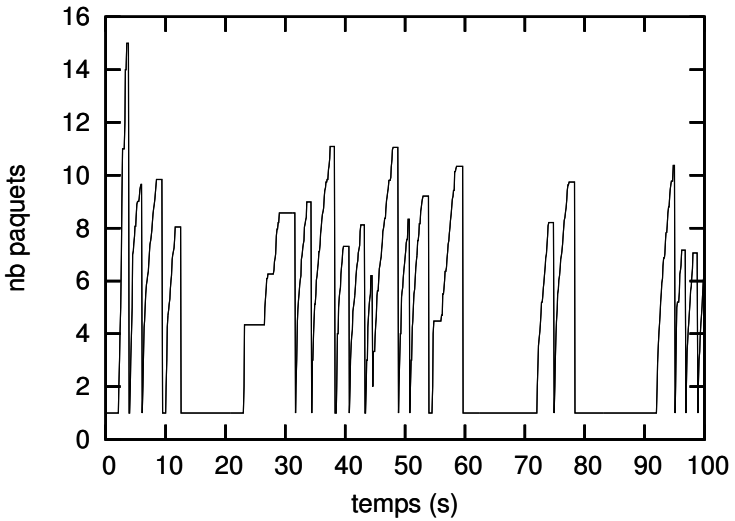
➔ propriété de régulation non assurée ?

➔ ou comportement inhérent au médium ?

TCP ◌ Ad hoc - mauvaise régulation

Nombreux signes de congestion ▷ la phase de départ lent de TCP est activé constamment ▷ la congestion reste : TCP est inefficace





Problèmes de stagnation : Reno ? ➔ oui, mais toujours un redépart en *Slow Start* toutes les secondes ➔ légère amélioration

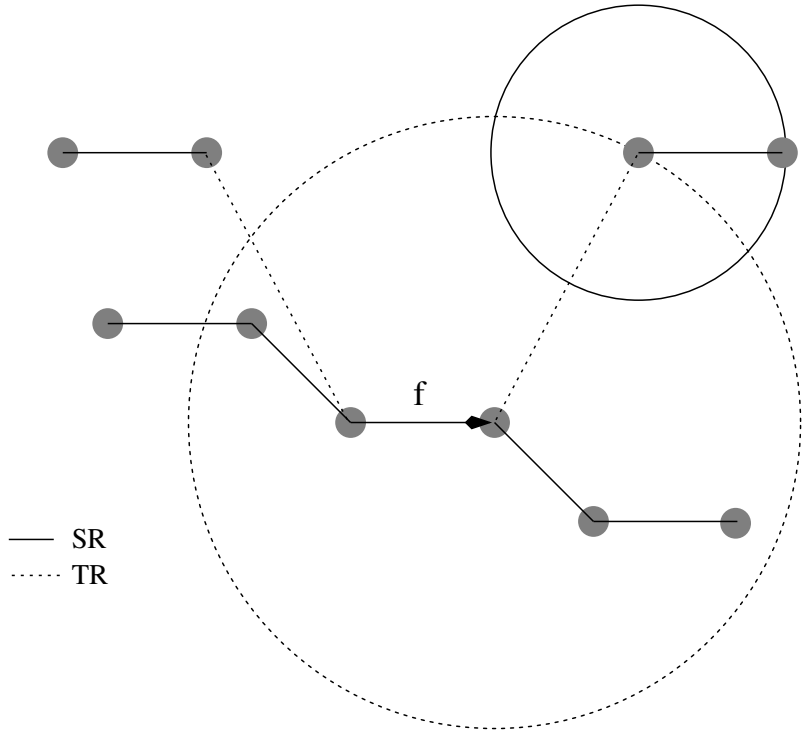
SACK, ECN ? ➔ légère amélioration, parfois non !

Ad hoc - l'accès au médium

En radio, collisions indétectables : paquets ACK et de réservation RTS/CTS

➔ coût de prise du médium

➔ large espace de compétition entre flux



Ad hoc - l'ensemble de congestion ad hoc (1)

Difficulté sur le site de Sally Floyd (Reno, SACK, ECN, RED, . . .)

> *If anyone has (partial) answers to any of these questions, I would be very interested.*
>
> *1- where is the congestion in the Internet ?*

Les buffers ne sont pas atteints. On va regarder les facteurs de pertes

Mobilité

– déconnexions : reconfiguration, pertes

  ➔ *handoffs* : minimiser les pertes (*smooth handoff*), minimiser la durée (*fast handoff*)

Ad hoc - l'ensemble de congestion ad hoc (2)

Routage

– construction des tables de routage ▷ blocage du médium : évaluation du RTT, déclenchement RTOs
– pertes de routes pendant la reconstruction : destructions
– protocole multi-routes ▷ paquets *out-of-sequence* : RTT, RTO, ACKs DUP

➔ influence imprévisible de la couche réseau

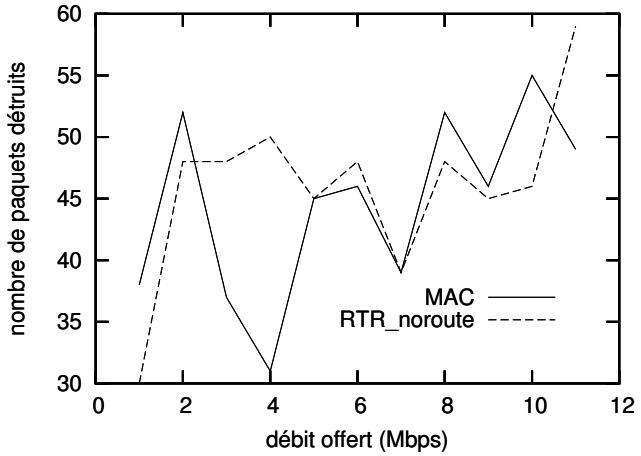
Médium

– fort taux d'erreur par bit (BER)
– retransmissions locales : RTT, concurrence avec la mécanique de fiabilité de TCP (ACKs DUP)

➔ nombreuses pertes, intéractions non triviales avec les niveaux supérieurs

TCP ○ Ad hoc - que faire ?

Constation TCP génère une congestion maximale, quel que soit le débit offert au niveau MAC (probabilité de destruction : 10^{-3})



3 voies

– informer, protéger les mécanismes de TCP de ces pertes
– améliorer les couches réseaux et liaison ⇝ ARQ/FEC
– adapater la régulation du débit de TCP à l'environnement ad hoc

Solutions - architectures inter-couches (1)

Idée garder la définition filaire de la congestion : différencier les pertes

➔ on bloque les variables de TCP (CWND, RTT, RTO) et on gèle les émissions le temps que les pertes s'arrêtent.

➔ on utilise des mécanismes *cross-layer* (les pertes viennent de différentes couches) : *feedbacks*, *cross-layer manager*, contrôle *hop-by-hop*, . . .

Mobilité, routage avec des paquets de signalisation, on prévient TCP et on le bloque. TCPBuS bufferise les paquets en transit afin de les retransmettre (+ réévaluation du RTO)

Signes de congestion protocole le plus finalisé : ATCP, une couche entre réseau et transport ▷ bloque les ACKs dupliqués, mémorisation des RTOs : il effectue lui même les restransmissions

Solutions - architectures inter-couches (2)

Proxys le protocole SplitTCP élit des proxys tout au long du parcours

▷ acquittements et régulation du débit entre proxys

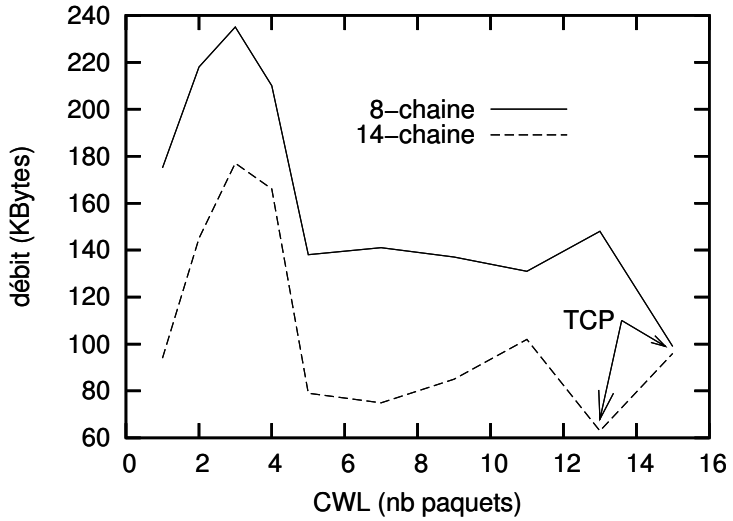
Problèmes

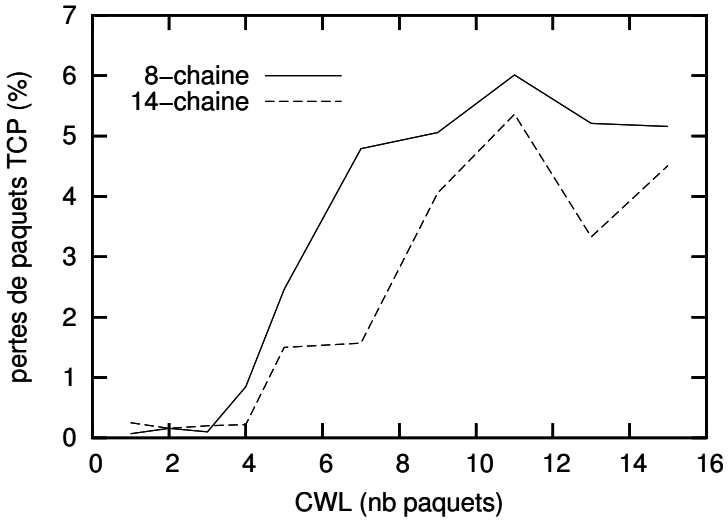
– compréhension du problème de plus en plus difficile
– perte de l'abstraction offerte par les couches
– surcharge par les paquets de signalisation
– performances très dépendantes des implémentations sujacentes

▷ Résultats réels et influences difficiles à évaluer

La borne *Congestion Window Limit*

Idée introduire peu de paquets dans le réseau : on borne la valeur de CWND avec CWL ▷ amélioration du débit d'au moins 50 %





On évite l'ensemble de congestion : probabilité de destruction 10^{-4}

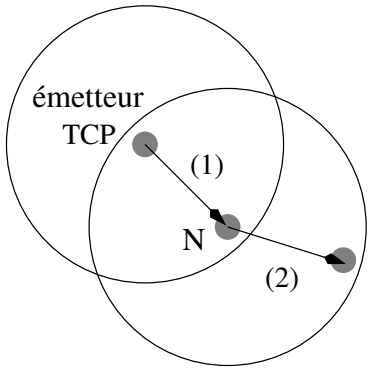
Réutilisation spatiale ie maximisation des émissions parallèles

comment fixer CWL ? (h=# liens radios) ▷ $\frac{h}{4}$ puis $\frac{2h}{5}$ ... impossible ! ▷ $c + f(h)$

TCPToK - contrôle du débit

➔ réguler CWND et oublier le mécanisme SS/CA

On va se baser sur la compétiton d'accès pour déterminer CWND



Envoi d'un paquet ▷ on sera exposé 2 fois : forward du paquet par N (faible mobilité) et à l'arrivée de l'acquittement

On va échanger des jetons avec notre voisinage et CWND va suivre leurs évolution ▷ # jetons = "ce qui est transférable dans le réseau"

Le nœud N devient notre indicateur de la réutilisation spatiale

TCPToK - contrôle du débit (algorithme)

Echange de jetons

- -1 jeton à chaque envoi
- +1 jeton récupérer à chaque réception du forward de N

Arrivée d'un ACK

- CWND = # jetons ➔ augmentation phase CA
- CWND > # jetons // ie la dernière fenêtre n'a pas été transféré
  - ➔ CWND = jetons
  - ➔ jetons = CWND - (lastCWND - jetons)

TCPToK - contrôle de la congestion

➔ utiliser CWL en cas d'apparition de congestion

▷ permet de diminuer significativement le nombre de paquets ▷ on veut une valeur de CWND petite, la réinitialisation n'aurait pas de sens

Algorithme

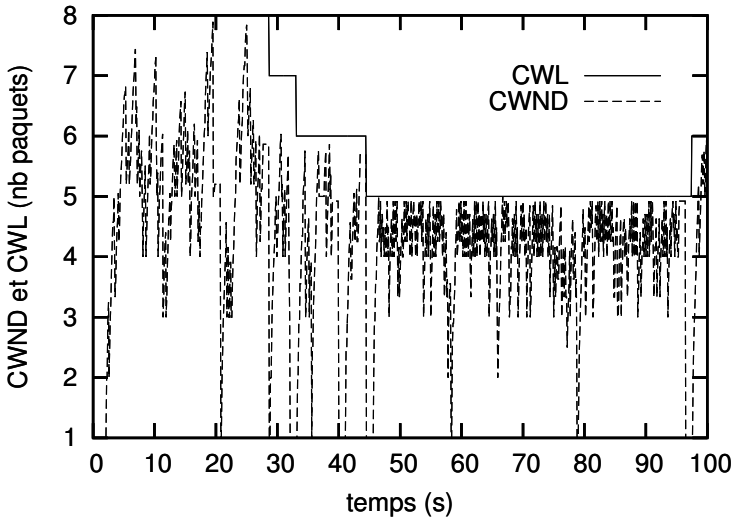
– timeout ou ACK dupliqué ➔ CWL = averageCWND (1 + α(lastCWL))
– arrivée d'un ACK et après stabilisation ➔ CWL = CWL + β averageCWND

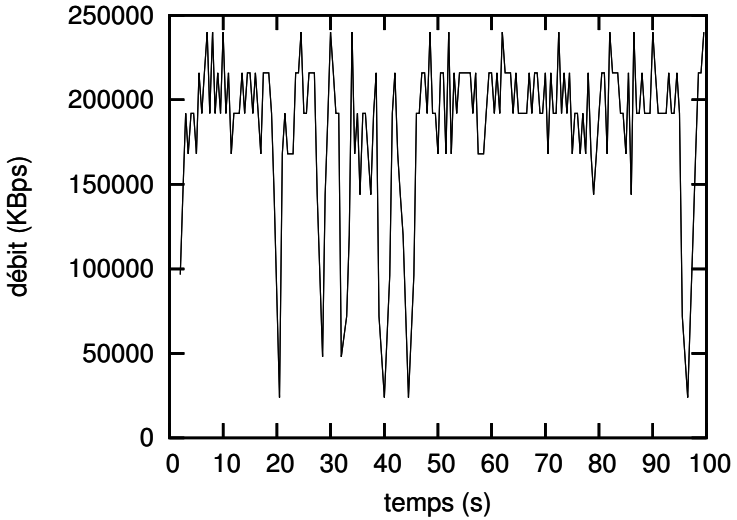
Au final

– On a séparé la régulation du débit et le traitement de la congestion
– On se base sur des indices inhérents au ad hoc et non sur des implémentations des couches inferieures.

TCPToK - résultats (1)

évolution de CWND et du débit dans une chaîne à 14 nœuds





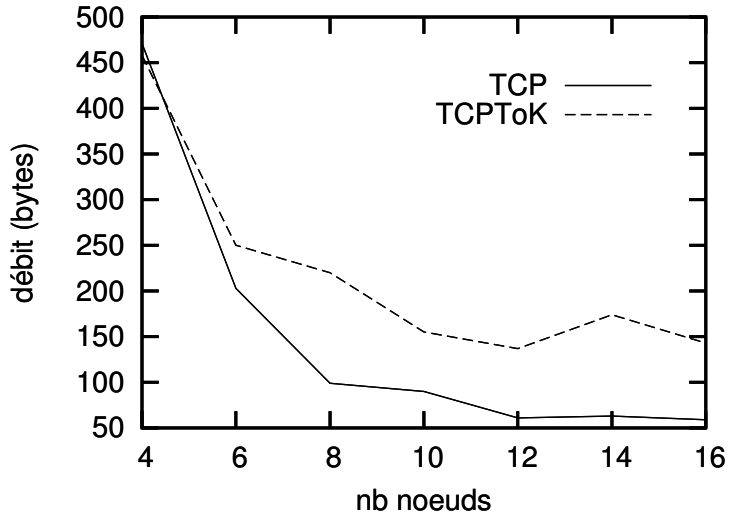
L'algorithme tend vers une régulation du débit idéale pour le réseau

▷ peu de pertes (0,25%) ▷ stabilité du débit : pas de gigue

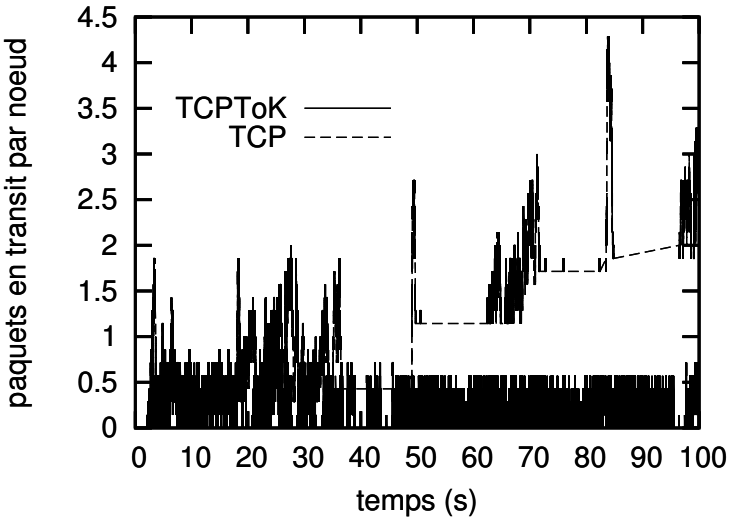
▷ la connexion améliore son débit de 35 à 200 %

TCPToK - résultats (2)

débit selon la longueur de la chaine



moyen de paquets aux nœuds



Avec TCPToK Reno ▷ on obtiendrait un comportement normal par rapport à la longeur de la chaîne (en 10, 12 et 16)

Nombre constant de paquets dans le réseau ▷ réutilisation spatiale réelle

TCPToK - résultats (3)

Dans toutes les configurations TCPToK améliore notablement le débit et le comportement :

	débit TCP (KBps)	débit TCPToK (Kbps)
(4,4)-grille 1 flux diagonal	171	200 (+17%)
(4,4)-grille 2 flux diagonaux	97	131 (+35%)
8-chaîne 4-goulot	139	187 (+35%)
12-chaîne 8-goulot	86	131 (+52%)
(12,4)-croix au noeud 6	(12-chaîne) 63 (4-chaîne) 208	72 (+14%) 271 (+30%)
8-chaîne handoff 7s	73	154 (+111%)

Conclusion

Nous avons

- caractérisé le comportement de TCP en ad hoc
- listé l'ensemble des problèmes qu'un protocole de transport rencontre en ad hoc
- mis au point une solution simple qui répond aux propriétés de TCP, adaptative et aux résultats très positifs rendant ce protocole viable en environnement ad hoc

Nous allons

- prendre en compte les acquittements $\leadsto$ DELHoC
- résoudre les problèmes de handoff, destruction, retransmission

¿ Questions ?

DELHoC

Idée au niveau des nœuds routeurs on utilise également un système de jeton mais réguler par l'argent

– coût de l'envoi : 1 unité

– réception d'un paquet : achat au prix α ($<$1)

– réception d'un jeton : achat au prix 1-α (inséré dans le paquet)

Au début α est à 1, puis selon si les jetons sont revenus α varie

$$\alpha = \frac{\#jetons\ recus}{\#jetons\ envoyes}$$

Lorsque l'on n'est plus assez riche pour envoyer un paquet on le détruit

Utilisation

➔ mécanisme ECN pour signifier la congestion

➔ prise en compte des acquittements dans TCPToK