RAPPORT DE RECHERCHE

L R I

I2SWP : ROUTING ALGORITHM WITH INTRA-FLOW INTERFERENCE CONSIDERATION IN AD HOC NETWORK

BENFATTOUM Y /MARTIN S /GAWEDZKI I /AL AGHA K

Unité Mixte de Recherche 8623
CNRS-Université Paris Sud – LRI

12/2010

Rapport de Recherche N° 1539

CNRS – Université de Paris Sud
Centre d'Orsay
LABORATOIRE DE RECHERCHE EN INFORMATIQUE
Bâtiment 490
91405 ORSAY Cedex (France)

I2SWP: Routing algorithm with intra-flow interference consideration in ad hoc network

Y. Benfattoum, S. Martin, I. Gawedzki, K. Al Agha

Laboratoire de Recherche en Informatique, Bât 490,
Université Paris-Sud 11 - CNRS F-91405, Orsay, France

Abstract

This paper presents a new routing heuristic for Quality of Service (QoS) in ad hoc networks, namely I2SWP, that addresses the issue of interference. In wireless networks, the medium is shared by many links therefore a new flow is subject to inter-flow interference caused by the existing flows and intra-flow interference caused by the flow itself. There are two main approaches that take into account the impact of a new flow on both types of interference. On the one hand, the cross-layer-based approach assumes that the network layer can recover statistics from the MAC layer, however this is not provided by currently available devices. On the other hand, the ASWP algorithm proposes a solution based only on the network layer but it cannot be implemented in real networks because of its high complexity. Our solution is inspired from the Shortest-Widest-Path algorithm with the addition of constraints to consider the interference while keeping a low complexity. Simulation results show that our approach is a good compromise between efficiency and complexity.

Résumé

Cet article présente une nouvelle heuristique de routage pour la qualité de service dans les réseaux ad hoc, à savoir I2ASWP, qui traite le problème des interférences. Dans les réseaux sans fil, le support de transmission est partagé par plusieurs liens et ainsi un nouveau flux peut être sujet à des interférences inter-flux causées par les flux existants et intra-flux causées par le flux lui même. Il y a deux approches principales qui prennent en compte l'impact du nouveau flux sur les deux types d'inteférences. D'une part, l'approche basée sur le cross-layering suppose que la couche réseau peut récupérer des statistiques de la couche MAC. Or, ceci n'est pas disponible sur les équipements actuels. D'autre part, l'algorithme ASWP, propose une solution basée uniquement sur la couche réseau mais ne peut être implémenté dans un cas réel à cause de sa complexité élevée. Notre solution s'inspire de l'algorithme du plus court chemin avec l'intégration de contraintes prenant en compte les interférences tout en maintenant une complexité raisonnable. Les résultats de simulation montrent que notre approche représente un bon compromis efficacité-complexité.

Key words: ad hoc network, routing, Quality of Service, intra-flow interference, admission control.

1 Introduction

A wireless ad hoc network is a decentralized network without a preexisting infrastructure. Each node participates in routing data of other nodes in addition of transmitting and receiving its own data. The routing process consists on determining the path along which the nodes will forward the data packets from a source to a destination node. In wireless networks, a node shares the medium with all its neighbors. Even if two paths have no node in common, interference can occur between them consequently reducing their throughput. This is known as inter-flow interference. Besides, when a link interferes with another link of the same path, the interference is said to be intra-flow.

In a wireless ad hoc network in which flows have bandwidth requirements, it is necessary to take into account this interference during the routing process. Indeed, if the network depicted on Figure 1 were wired, it is clear that the available bandwidth for a flow from a to c would be equal to the least of the bandwidths of $(a \rightarrow b)$ and $(b \rightarrow c)$. Because the network is wireless, the available bandwidth must be divided by two since both considered links cannot transmit simultaneously.

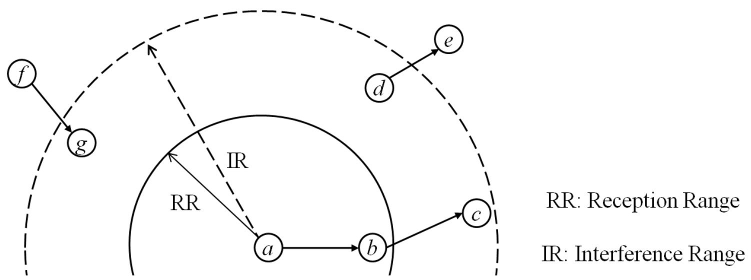

Fig. 1. Intra-flow and inter-flow interference.

Most of the routing algorithms with QoS (Quality of Service) do not take into account the interference caused by the new flow with itself. Consequently, the new flow can be accepted while the required bandwidth is not available on the chosen path. Some algorithms consider these intra-flow interference by assuming a cross-layer model, a N-hops interference model or clique interference model.

In this paper, we propose the Intra-flow Interference-aware Shortest-Widest-Path (I2SWP), a low complexity solution to find a feasible path in terms of bandwidth for a new flow, by considering its intra-flow interference. The cross-layer model is not used in the available devices and the N-hops interference model is approximate. The computation of the cliques in a conflict graph is an interesting way to find the exact sets of mutually interfering links. This model

Email address: {youg, smartin, ig, alagha}@lri.fr (Y. Benfattoum, S. Martin, I. Gawedzki, K. Al Agha).

and the problem are detailed in Section 2. Section 3 presents the state of the art for the routing algorithms that partially solve the considered problem. In Section 4, we propose improvements of existing solutions. We show by simulation, in Section 5, that our solution is a good compromise between efficiency and complexity. Finally, we conclude in Section 6.

2 Routing and Interference

We are interested in the issue of routing in a wireless ad hoc network in which flows have constraints in term of bandwidth. More exactly, when a new flow is introduced in the network, it must avoid creating inter-flow interference that can reduce the bandwidth reserved by the already accepted flows. Furthermore, the impact of the new flow interference on itself (intra-flow interference) must be taken into account to know if the considered path can provide the desired bandwidth or not. This last point is rarely handled by the existing routing algorithms.

2.1 Interference and Widest-Path

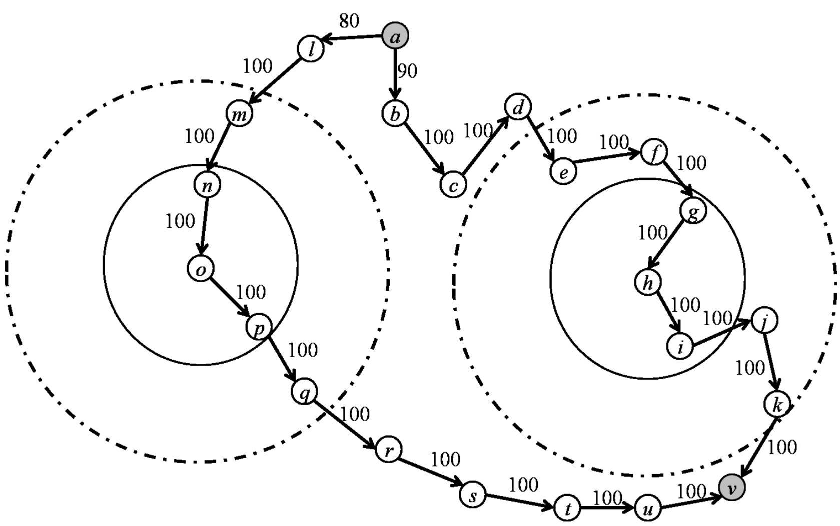

Fig. 2. Finding a path with interference consideration

Let's take for example Dijkstra's widest-path algorithm: since the path width is equal to the least bandwidth on its links, we can see on Figure 2 that the algorithm is not optimal when used in wireless networks. Indeed, there are two paths from a to v with the same length. The weights of the links represent their available bandwidth. The path chosen by Dijkstra's algorithm is on the right because the smallest bandwidth is equal to 90, versus 80 for the one on the left. However, the intra-flow interference is more important on this path. For example, when the link $(h \to i)$ is transmitting, the seven links $(d \to e)$, $(e \to f)$, $(f \to g)$, $(g \to h)$, $(i \to j)$, $(j \to k)$ and $(k \to v)$ should be inactive. Along the left path, when $(o \to p)$ is transmitting, only six links $(l \to m)$, $(m \to n)$, $(n \to o)$, $(o \to q)$, $(q \to r)$ and $(r \to s)$ should be inactive. If the bandwidth required by the flow is equal to 14, then the chosen path is not valid, because the real bandwidth cannot exceed $100/(7+1) < 14$. On the other hand, the left path satisfies the request of the flow because the real bandwidth is $100/(6+1) > 14$. We propose in Section 4 improvements to cope with the issue of intra-flow interference of the new flow.

2.2 Mutually Interfering Links

We consider a network as a graph $G(V, E)$, with V and E the sets of nodes and links respectively. The conflict graph CG is an undirected graph in which the vertices represent links of the network G and the edges interference relations between links. Figure 3 shows an example of five nodes in which

- the link $(a \to b)$ interferes with links $(a \to c)$ and $(b \to c)$
- the link $(b \to c)$ interferes with links $(a \to b)$, $(a \to c)$ and $(c \to d)$
- the link $(a \to c)$ interferes with links $(a \to b)$, $(b \to c)$, $(c \to d)$ and $(d \to e)$
- the link $(c \to d)$ interferes with links $(a \to c)$, $(b \to c)$ and $(d \to e)$
- the link $(d \to e)$ interferes with links $(a \to c)$ and $(c \to d)$.

We represent a set of mutually interfering links by a maximal clique in the conflict graph (i.e., a complete sub-graph of CG). In this example, there are three maximal cliques. Finding all maximal cliques[1] in a graph is an NP-complete problem and the most efficient algorithm for finding all the cliques [3] is linear according to the number of cliques.

There are other methods to determine the sets of mutually interfering links in a network, some based on a model of N-hops interference, others on cross-layer approaches. The first category gives an approximation of the cliques in the conflict graph, whereas the second uses information from the MAC layer to find interfering nodes [12]. However, the currently available off-the-shelf devices assume the use of a TCP/IP stack and thus do not allow cross-layering. To be effective, a node applying a routing algorithm with QoS in terms of bandwidth should have a local knowledge of its links but also a global view of the sets of mutually interfering links to which its links belong.

---

[1] We shall use the term "clique" instead of "maximal clique" in the following, for simplicity's sake.

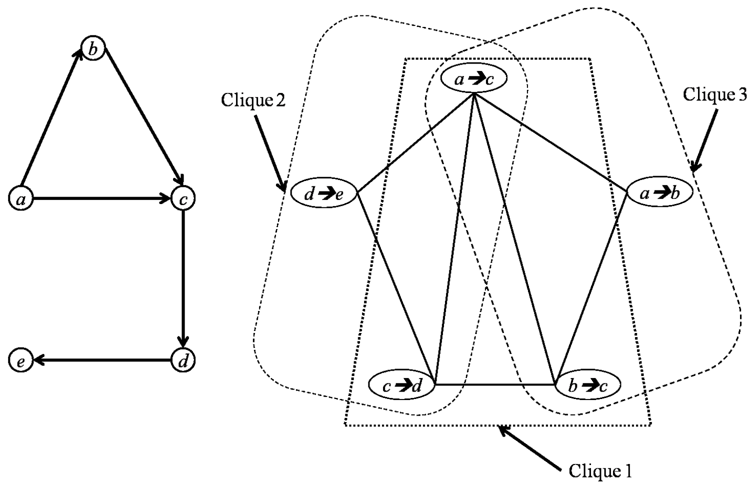

Fig. 3. Violation of the optimality principle in a wireless network (example taken from Jia's paper).

Besides the computation of the cliques, finding the optimal path from a source to a destination node is also NP-complete. Indeed, it has been shown by Jia *et al.*[6] that the principle of optimality is not respected in a wireless network. In other words, the best path from a source to a destination is not necessarily the best one from the source to an intermediate node due to the interference. Assuming that all the links of Figure 3 have the same capacity C, the widest path from node a to node c is $a \to c$. On the other hand, the widest path from a to e is $a \to b \to c \to d \to e$ because the available bandwidth $C/2$ is larger than that of the path $a \to c \to d \to e$ ($C/3$). Thus the best path towards the destination is not the succession of the best paths towards intermediate nodes. This explains the high complexity of the problem.

Indeed, a comprehensive solution consists in testing all the possible paths from the source to the destination. The following section presents the existing routing algorithms. As we shall explain, these are either insufficient or have a high complexity. We shall present in Section 4 some improvements to satisfy the requirements of the flows while keeping a relatively low complexity.

3 State of The Art

We are not interested in cross-layer based algorithms. These use information from the MAC layer such as the idle channel time in order to recover some parameters necessary for routing [12,9]. Most existing architectures are not adapted to use cross-layer methods [8].

In the following, we present the main routing algorithms based only on the network layer, namely Dijkstra, Shortest-Widest-Path (SWP), the algorithm based on Integer Linear Programming (ILP), Ad hoc SWP (ASWP) and its variant I2ASWP. Each of these algorithms searches the best path from a source s to a destination d, by using different criteria:

Dijkstra	The best path is the widest in term of bandwidth.
SWP	The best path is the shortest having a bandwidth greater or equal to what is required by the flow.
ILP	The best path is the shortest in terms of the number of hops and it is possible to add other constraints to respect the QoS requirements.
ASWP	The best path is the widest in terms of bandwidth with intra-flow and inter-flow consideration during the width computation.

3.1 Dijkstra

The Dijkstra algorithm [4] finds the shortest path from a source to a destination in a weighted graph. Its advantage is that it allows to find the shortest paths from a source to all the vertices in the graph in a single run. It is however possible to change the criterion of minimization of the path length to maximization of the smallest residual bandwidth on the path. The residual bandwidth of a link $(u \to v)$ is defined as follows:

$$S(u \to v) = C(u \to v) - R(u \to v) \ , \tag{1}$$

with $C(u \to v)$ the initial bandwidth of $(u \to v)$ and $R(u \to v)$ the bandwidth already reserved by the existing flows on this link. Let $path_{s,v}$ be a path from s to v, its width is given by:

$$W(path_{s,v}) = \min_{(i \to j) \in path_{s,v}} S(i \to j) \ . \tag{2}$$

This algorithm finds a feasible path in term of bandwidth with a polynomial complexity, but the chosen criterion (the residual bandwidth) does not consider interference. A redefinition of the residual bandwidth will be given in Section 4 in order to adapt the algorithm to our problem.

3.2 SWP

The previous algorithm can find relatively long paths. A long path consumes more resources and causes more interference in the network. The idea of SWP

[11] is to keep only the links having a residual bandwidth greater than the bandwidth required by the new flow. So, it considers only feasible paths in terms of bandwidth. The SWP method then applies Dijkstra's algorithm minimizing the length of paths.

This algorithm gives a better solution than Dijkstra's to solve our problem but the interference is still not taken into account.

3.3 ILP

The shortest path algorithm can be modeled as a linear programming problem that contains an objective function to minimize (or maximize) and a set of constraints using integer variables. In this case it is called Integer Linear Programming [10].

To each unidirectional link $(u \rightarrow v)$ of the topology is associated an integer variable $x_{(u \rightarrow v)}$. The problem to solve is then

$$\text{Minimize} \sum_{(u \rightarrow v) \in E} x_{(u \rightarrow v)}$$

s.t.

$$\begin{cases} \sum_u x_{(u \rightarrow v)} - \sum_w x_{(v \rightarrow w)} = 0 \ , & \forall v \in V \setminus \{s, d\} \ , \\ & \forall (u \rightarrow v), (u \rightarrow w) \in E \\ \sum_u x_{(u \rightarrow s)} - \sum_w x_{(s \rightarrow w)} = -1 \ , & \forall (u \rightarrow s), (s \rightarrow w) \in E \\ \sum_u x_{(u \rightarrow d)} - \sum_w x_{(d \rightarrow w)} = 1 \ , & \forall (u \rightarrow d), (d \rightarrow w) \in E \\ \qquad x_{(u \rightarrow v)} \in \{0, 1\} \ , & \forall (u \rightarrow v) \in E \end{cases}$$

In a solution of this problem, the variables $x_{(u \rightarrow v)}$ that are equal to 1 indicate the links of the best path from s to d.

This algorithm is not used in practice because a linear program solver has a high complexity. Its advantage is that it allows to integrate easily more constraints offering guarantees of QoS on several criteria (e.g., bandwidth, delay and loss rate).

3.4 ASWP

The Ad hoc Shortest-Widest-Path (ASWP) was proposed by Jia *et al.* [6], it takes into account both kinds of interference and uses the clique interference model. This heuristic is based on the Bellman-Ford algorithm [2],[7] combined to k-best-paths approach. Indeed, each node keeps locally the k widest paths from the source to it. When k is equal to the number of feasible paths from the source to the destination, the result of the algorithm is exactly the widest path considering interference. A lower k gives approximate results.

The authors assume a constant channel capacity C. They define the capacity of a clique as follows

$$c_q = \alpha \cdot C - \sum_{(u \to v) \in q} R(u \to v) \ , \tag{3}$$

with $\alpha \approx 0.46$. This was demonstrated by Gupta *et al.* [5].

The width of a path from s to v is given by

$$W(path_{s,v}) = \min_{q \in Q} \frac{c_q}{z_q^{s,v}} \ , \tag{4}$$

with $z_q^{s,v}$ the number of links of the path from s to v which belong to the clique q. This algorithm calculates the exact width of paths from s to all the destinations considering the new and existing flow interference.

Consequently, ASWP is suitable for our problem, but it has a relatively high complexity of $o(k \cdot |Q| \cdot |V| \cdot |E|^2)$ with $|V|$, $|E|$ and $|Q|$ respectively the number of nodes, links and cliques.

When the number of nodes becomes high, this algorithm is not applicable due to its complexity. In the following section, we propose a solution that has a good efficiency while keeping a lower complexity.

4 Our Solution

The algorithms presented in the previous section do not take into account neither inter-flow nor intra-flow interference. While ASWP solves correctly our problem, it does that at a high cost. In this section, we propose some improvements to address these problems. A comparison among the different solutions is presented in the next section.

When introducing a flow in a network, the routing algorithm must find a path on which:

- the interference generated by the new flow on those already accepted must not degrade their guaranteed bandwidths;
- the interference of the new flow on itself must not prevent it from having the required bandwidth.

A redefinition of the link residual bandwidth by integrating clique constraints solves the first point. Let r_q be the utilization rate of a clique q. It is the sum of utilization rates of all its links

$$r_q = \sum_{(u \to v) \in q} \frac{R(u \to v)}{C(u \to v)} \ , \tag{5}$$

where $R(u \to v)$ is the bandwidth reserved on the link $(u \to v)$ by the existing flows and $C(u \to v)$ the initial bandwidth of this link.

The residual bandwidth of a link $(u \to v)$ is then a function of the highest utilization rate of the cliques to which the link belongs. It is defined as follows:

$$S(u \to v) = C(u \to v) \cdot \max\left(1 - \max_{\substack{q \in Q \\ (u \to v) \in q}}(r_q), 0\right) \ . \tag{6}$$

The second point implies to redefine the width of a path by considering the intra-flow interference of the new flow. We give a formula inspired by the ASWP approach to compute the width of a path. In ASWP, the channel capacity is assumed to be constant. For a variable channel capacity, we adapt the computation of the width as follows: let $path_{s,v}$ a path from s to v. Its width, noted $width_{s,v}$, is a function of residual bandwidths divided by a quantity $z_q^{s,v}$ that is the number of links of the considered path which belong to the clique q.

$$width_{s,v} = \min_{q \in Q}\left(\frac{1 - r_q}{z_q^{s,v}} \cdot \min_{(i \to j) \in q \cap path_{s,v}} C_{(i \to j)}\right) \tag{7}$$

N-Hops Interference Model

The computation of the residual bandwidths is done only once before running the algorithms. On the other hand, the width of the path is computed in Dijkstra's and ASWP algorithms at every relaxation. This computation is based on an exponential number of cliques and doing it many times iteratively to find a path highly increases the complexity.

To cope with it, we compute the path width by using a well-known approach: the n-hops interference model. It supposes that a node interferes only with its n-hop neighbors (generally $n = 2$). Figure 4 shows a scenario of communication in which the transmitting (resp. idle) links are represented with continuous (resp. dotted) lines. Two links of the same clique cannot transmit simultaneously. According to that example, only one link in four can be in transmission mode at once.

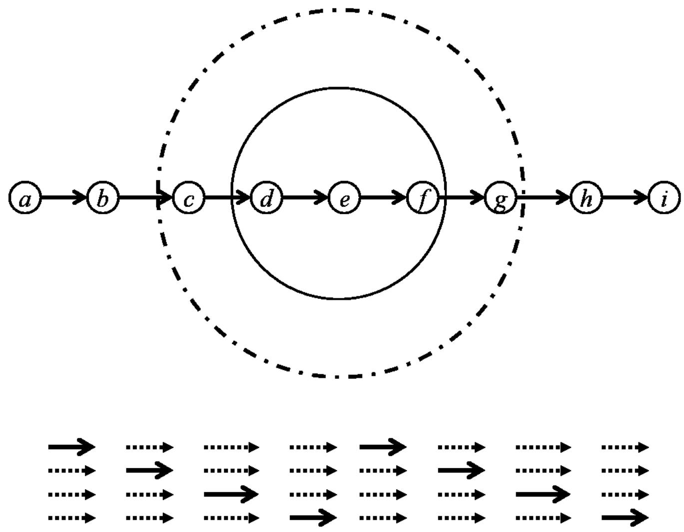

Fig. 4. Representation of intra-flow interference in a n-hops interference model.

The bandwidth of a flow going from s toward v along a path of length $len_{s,v}$ can be approximated by the smallest bandwidth on this path divided by $len_{s,v}$ if $len_{s,v} \leq 4$ or by 4 otherwise.

When an algorithm is based on an n-hops model, the interference is not taken into account in an exact way. Indeed, in Figure 2, the bandwidth of the link $(h \rightarrow i)$ interferes with seven other links. The bandwidth would be $100/(7+1)$ while with the n-hops model, it would be $100/4$.

In that case, the utilization rate of a clique can exceed 100% and we call this phenomenon *clique violation*. Consequently, flows of the network may see their allocated bandwidth decreased. An admission control mechanism is thus necessary to verify that the path from s to d found by the algorithm has a width larger or equal to the bandwidth required by the flow, otherwise the flow is rejected.

Our improvements aim to find a good compromise between efficiency and complexity. We take the algorithms seen in Section 3 one by one and adapt them to our problem by integrating the new definition of the residual capacity and the approximation of the n-hops model seen above.

4.1 I2Dijkstra

A possible improvement of Dijkstra's algorithm from Section 3 is that every node v keeps track of the path by which it is reached from the source s. Every node can then compute the width of its path such as defined in (7), and use it as a weight in the algorithm.

However, the complexity of this solution is high because the path width is computed on every relaxation.

We propose the Intra-flow Interference-aware Dijkstra algorithm (I2Dijkstra) which improves Dijkstra's algorithm by considering the intra-flow interference constraints obtained with the n-hops interference model.

At the end of this algorithm, the path found has a bandwidth approximated by the weight of the destination d divided by α_d.

4.2 I2SWP

To consider the inter-flow interference, the Intra-flow Interference-aware Shortest-Widest-Path (I2SWP) algorithm applies a Dijkstra shortest-path algorithm to the graph G deprived of all the links having a residual bandwidth (such as defined in (6)) lower than the bandwidth B required by the new flow.

It is possible that the chosen shortest path violates the cliques constraints, because the intra-flow interference is not taken into account. For that reason, in the admission control step (downstream from the I2SWP), the chosen path, noted $path_{s,d}$, is tested for the following constraint

$$\forall q \in Q, \quad r_q + \sum_{\substack{(u \to v) \in q \\ path_{s,d}}} \frac{B}{C(u \to v)} \leq 1 \ . \tag{8}$$

If the constraints are not satisfied, which means at least one clique is overloaded, the flow is rejected, otherwise it is accepted. Note that when this algorithm finds a path, it is feasible, whereas the reverse is not true. Indeed, if the algorithm does not return any path, it could be that a longer path is valid with respect to the constraints (because it goes through cliques that are less loaded). The advantage of this algorithm is a relatively low complexity of $o(|V|^2 + |Q| \cdot |E|)$.

4.3 I2ILP

For the ILP algorithm, we suggest to integrate clique constraints in the initial problem. The final algorithm, namely Intra-flow Interference-aware ILP (I2ILP), adds the following constraints to the original one

$$\forall q \in Q, \quad r_q + \sum_{(u \to v) \in q} \frac{B \cdot x_{(u \to v)}}{C(u \to v)} \leq 1 \ . \tag{9}$$

which can be written as follows

$$\forall q \in Q, \quad \sum_{(u \to v) \in q} \frac{B \cdot x_{(u \to v)} + R(u \to v)}{C(u \to v)} \leq 1 \ . \tag{10}$$

Unlike I2SWP, the I2ILP always gives a feasible path if there is one in the network.

4.4 I2ASWP

To reduce the high complexity of the ASWP, which is in $o(k \cdot |Q| \cdot |V| \cdot |E|^2)$, we have suggested in the Intra-flow Interference-aware ASWP (I2ASWP) [13] to approximate the estimation of the intra-flow interference by using the n-hops model. Let $len_{s,v}$ be the length of the path from s to v. The width of $path_{s,v}$ is given as follows

$$W_{s,v} = \begin{cases} \min\limits_{(i \to j) \in path_{s,v}} S(i \to j) \Big/ len_{s,v} & \text{if } len_{s,v} \geq 4 \\ \min\limits_{(i \to j) \in path_{s,v}} S(i \to j) \Big/ 4 & \text{otherwise} \ . \end{cases} \tag{11}$$

This improvement allows to reduce significantly the complexity, since I2ASWP has a complexity of $o(k \cdot |V| \cdot |E| + |Q| \cdot |E|)$.

5 Performance Evaluation

In the chosen model, the network is connected and the nodes are fixed (no mobility). The links are symmetric: the initial bandwidth of a link $(a \to b)$ is equal to that of the link $(b \to a)$. The channel capacity is supposed variable, in other words, the links can have different initial bandwidths.

Our simulator, written in C++, implements only the network layer. We use the OPNET scheduler to manage the events of starting and finishing the flows over time. The network consists of a connected graph of 30 nodes randomly placed on a surface of 20×20 units. The reception range radius (RR) is set to 4 units and the interference range radius (IR) to 8 units. The initial bandwidth of a link is randomly chosen between 500 and 1000. Many flows are submitted to the network, each of them starts from a source node s to a destination node d randomly chosen, and requires a QoS consisting of a bandwidth B according to an exponential distribution of rate 80^{-1}. The flow interarrival and lifetime follow exponential distributions of rate 500^{-1} and 5^{-1} respectively.

The algorithms as of Section 4 are implemented with the help of the GLPK linear programming solver [1] for I2ILP.

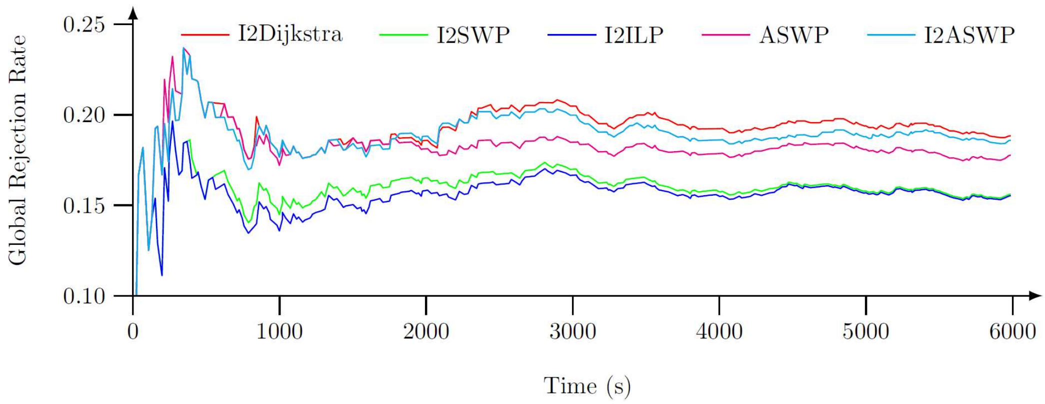

Fig. 5. Evolution of global rate of rejection over time.

The evolution of the global rejection rate over simulation time is presented in Figure 5, for each algorithm. Until 500 s, the low load in the network makes the rejection rates very variable, but soon after, as the load reaches its peak, the rejection rates become stable in relation to each other.

It appears the best results are obtained with the theoretical algorithm I2ILP and the I2SWP. Indeed, these two aim to find a satisfactory path which should be relatively short. Consequently, they use less links and less cliques leading to increase the ability to accept new flows.

The ASWP has a higher rejection rate. It finds the exact widest path by taking into account the interference of the existing and the new flows. Since the chosen is usually wider than strictly required, the residual capacity of the links on the path (after insertion of the new flow) is important. This has a beneficial effect on the ability to accept new flows.

The I2ASWP and I2Dijkstra have the highest rates of rejection.

Table 1
Average proportion of violated cliques during the simulation

I2Dijkstra	**I2SWP**	**I2ILP**	**ASWP**	**I2ASWP**
0.0098	0	0	0	0.0118

Table 1 shows the average proportion of the overloaded cliques during the simulation. It is the ratio between the number of cliques having exceeded a capacity of 100% and the total number of cliques.

It appears that only both approaches I2ASWP and I2Dijkstra violate clique constraints quite often because they do not consider the intra-flow interference in an exact way. When a clique is overloaded, all its links are unusable because their residual capacities are set to zero. Then it becomes more difficult to accept new flows. Consequently, the number of admitted flows with these two approaches is lower than that of the others.

Table 2
Average path length and load of admitted flows

	I2Dijkstra	I2SWP	I2ILP	ASWP	I2ASWP
Length	4.48	3.91	3.93	4.22	4.55
Load	61.10	63.03	63.58	59.36	62.28

Table 2 shows that I2Dijksta and I2SWP select paths longer than those selected by the other algorithms. Conversely, I2ILP and I2SWP select the shortest paths. They also admit flows with a higher average required bandwidth.

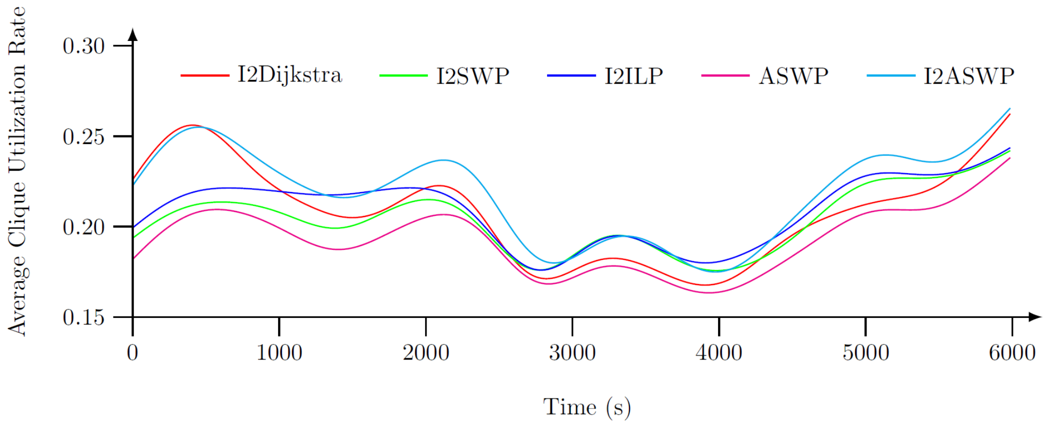

Fig. 6. Average clique utilization over time.

The evolution of average clique utilization rate is presented in Figure 6. The curves have been low-pass filtered for better legibility of the global trends. It appears that ASWP is the one using the least clique capacities. Despite of having a higher acceptation rate, I2SWP and I2ILP have a lower clique utilization rate compared to I2Dijkstra and I2ASWP.

Consequently, the I2SWP seems to be the best compromise to guarantee:

- a high acceptation rate;
- least clique violation;
- a low complexity of $o(|V|^2 + |Q| \cdot |E|)$.

6 Conclusion

In wireless ad hoc networks, the routing algorithms guaranteeing QoS in term of bandwidth must take into account the notion of interference during the routing process. Among the existing solutions, most do not consider neither the interference of the existing flows on one another nor of a flow on itself. For those that do take interference into account, either the computation cost is high or a cross-layer approach is required, which is not practical in currently available devices. We propose some improvements of existing algorithms in order to take the interference into account and showed by simulation that I2SWP is both efficient and has a relatively low complexity.

As a possible further improvement, it would be interesting to use a heuristic for the computation of the cliques, instead of an exact algorithm. Then of course addressing the impact of mobility would allow to investigate in a more general way the behavior of our routing algorithms.

References

[1] http://www.gnu.org/software/glpk/. Accessed on October 2010.

[2] Richard Bellman. On a routing problem. *Quarterly of Applied Mathematics* 16, 87-90 (1958).

[3] Coen Bron and Joep Kerbosch. Algorithm 457: finding all cliques of an undirected graph. *Commun. ACM* 16, 575-577 (1973).

[4] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik* 1, 269-271 (1959).

[5] Rajarshi Gupta, John Musacchio, and Jean Walrand. Sufficient rate constraints for QoS flows in ad-hoc networks. *Ad Hoc Netw.* 5, 429-443 (2007).

[6] Z. Jia, R. Gupta, J. Walrand, and P. Varaiya. Bandwidth guaranteed routing for ad hoc networks with interference consideration. In: Proc. IEEE Symposium on Computers and Communications. p. 3-9 (2005).

[7] L. R. Ford and D. R. Fulkerson. Flows in networks. Princeton University Press, NJ (1962).

[8] S. Odou, S. Martin, and K. Al Agha. Idle channel time estimation in multihop wireless networks. IEEE International Conference on Communications (ICC'04), Dresden, Germany (2009).

[9] K. Sanzgiri, I. D. Chakeres, and E. M. Belding-Royer. Determining intra-flow contention along multihop paths in wireless networks. In : Proc. of the First International Conference on Broadband Networks (BROADNETS'04), p. 611-620, Washington, DC, USA (2004).

[10] A. Schrijver. Theory of linear and integer programming. John Wiley & Sons,Inc., New York, NY, USA (1986).

[11] Z. Wang and J. Crowcroft. Quality of service routing for supporting multimedia applications. *IEEE Journal on Selected Areas in Communications*, 14, 1228-1234 (1996).

[12] Y. Yang and R. Kravets. Contention-aware admission control for ad hoc networks. *IEEE Transactions on Mobile Computing*, 4, 363-377 (2005).

[13] Y. Benfattoum, S. Martin, I. Gawedzki, and K. AlAgha. I2ASWP: Manet routing considering intra-flow interference. In : Proc. of 4th Asia-Pacific International Symposium on Advanced Reliability and Maintenance Modeling (APARM10), Wellington, NZ (2010).