Fast And Simple Approximation Algorithms for Maximum Weighted Independent Set of Links

Peng-Jun Wan*, Xiaohua Jia†, Guojun Dai‡, Hongwei Du§, and Ophir Frieder¶
* Department of Computer Science, Illinois Institute of Technology, wan@cs.iit.edu
† Department of Computer Science, City University of Hong Kong, csjia@cityu.edu.hk
‡ School of Computer Science, Hangzhou DianZi University, daigj@hdu.edu.cn. Corresponding author.
§ Department of Computer Science and Technology, HIT Shenzhen Graduate School, hwdu@hitsz.edu.cn
¶ Department of Computer Science, Georgetown University, ophir@ir.cs.georgetown.edu

***Abstract*—Finding a maximum-weighted independent set of links is a fundamental problem in wireless networking and has broad applications in various wireless link scheduling problems. Under protocol interference model, it is NP-hard even when all nodes have uniform (and fixed) interference radii and the positions of all nodes are available. On one hand, it admits a polynomial-time approximation scheme (PTAS). In other words, for any fixed $\varepsilon > 0$, it has a polynomial-time (depending on ε) $(1+\varepsilon)$-approximation algorithm. However, such PTAS is of theoretical interest only and is quite infeasible practically. On the other hand, only with the uniform interference radii is a simple (greedy) constant-approximation algorithm known. For the arbitrary interference radii, fast constant-approximation algorithms are still missing. In this paper, we present a number of fast and simple approximation algorithms under the general protocol interference model. When applied to the plane geometric variants of the protocol interference model, these algorithms produce constant-approximate solutions efficiently.**

I. Introduction

Consider a multihop wireless network with a set L of communication links. A set I of links in L is said to be *independent* if all links in I can transmit successfully at the same time under a pre-specified interference model. Given a subset A of L and a positive weight function w on A, the problem of finding an independent subset (abbreviated with IS) I of A with maximum total weight $\sum_{a\in I} w(a)$ is known as **Maximum Weighted Independent Set of Links** (**MWISL**). In particular, given a subset A of L, the problem of finding a largest IS of A is known as **Maximum Independent Set of Links** (**MISL**). The problem **MWISL** plays fundamental roles in many wireless link scheduling problems. For examples, Wan [5] presented polynomial approximation-preserving reductions from three wireless link scheduling problems minimum-latency link scheduling, maximum multiflow, and maximum concurrent multiflow to **MWISL**. In other words, if there exists a polynomial μ-approximation algorithm for **MWISL**, then there also exists polynomial μ-approximation algorithms for those three problems as well. Lin and Shroff [4] proved that for the maximum-throughput stable wireless link scheduling, any μ-approximation algorithm for **MWISL** also achieves a stable throughput efficiency ratio at least $1/\mu$.

Because of its fundamental importance, the problem **MWISL** received much research interest in the past decade. Most of the existing studies (e.g., [1], [2], [3], [4], [5]) assumed some variants of the *protocol* (as opposed to *physical*) interference model. In general, a protocol interference model specifies a pairwise conflict relations among all links in L, and a subset I of L is independent if all links in I are pairwise conflict-free. It is classified into two communication modes:

- Unidirectional mode: For each link $a=(u,v)\in L$, the communication between u and v occurs in the direction from u to v, and the endpoint u (respectively, v) is referred to as the sender (respectively, receiver) of a. The sender u of the link a has an interference range, and the interference range of a is the interference range of its sender. Two links in A conflict with each other if and only if the receiver of at least one link lies in the interference range of the other link.
- Bidirectional mode: For each link $a=(u,v)\in L$, the communication between u and v occurs in both directions, and both u and v have an interference range. The interference range of a is the union of the interference ranges of its two endpoints. Two links in A conflict with each other if and only if at least one link has an endpoint lying in the interference range of the other link.

In the plane geometric variant, the interference range of an endpoint u of a link a is assumed to be a disk centered at u of radius $r_a(u)$, which is also knows as the interference radius. Under the plane geometric variant of the protocol interference model in either unidirectional mode or bidirectional mode, the computational hardness of the problem **MWISL** was well characterized in [5]. On one hand, the problem **MISL** (and hence the problem **MWISL** too) is NP-hard even when all nodes have uniform (and fixed) communication radii and uniform (and fixed) interference radii and the positions of all nodes are available. On the other hand, the problem **MWISL** (and hence the problem **MISL** too) admits a polynomial-time approximation scheme (PTAS). In other words, for any fixed $\varepsilon>0$, it has polynomial-time (depending on ε) $(1+\varepsilon)$-approximation algorithm. Such PTAS is of theoretical interest only and is quite infeasible practically as it involves very time-consuming exhaustive enumerations. For practical approximation algorithms for **MWISL**, only a simple greedy 8-approximation algorithm [1] is known in the bidirectional mode with *uniform* interference radii, which selects an independent set of links in the first-fit manner in the decreasing order of link weights. In all other settings, practical approximation algorithms for **MWISL** are still missing till now.

In this paper, we present a number of fast and simple approximation algorithms for **MWISL** under the general protocol interference model, all of which exploit the rich nature of the protocol interference model. Under the plane geometric variant of the protocol interference model in either unidirectional mode or bidirectional mode, these approximation algorithms achieve constant approximation bounds even with arbitrary interference radii. In the same setting as in [1] (i.e., the bidirectional mode with uniform interference radii), we are able to achieve an approximation bound between 3 and 6 with our algorithms, which outperforms the 8-approximation bound achieved in [1].

The remainder of this paper is organized as follows. In Section II, we design and analyze a general orientation-based approximation algorithm, and study its performance under specific variants of the protocol interference model. In Section III, we design and analyze a general ordering-based approximation algorithm, and study its performance under specific variants of the protocol interference model. In Section IV, we develop a divide-and-conquer approximation algorithm, which is specially tailored for the plane geometric variant of the protocol interference model with uniform interference radii. We conclude this paper in Section V. The following standard terms and notations are adopted throughout this paper.

- $\mathcal{I}$ denotes the collection of all independent subsets of A.
- G denotes the link-conflict graph of A under the given protocol interference model. In other words, A is the vertex set of G and two links in A are adjacent in G if and only if they conflict with each other.
- For any $a \in A$, $N(a)$ denotes the set of links in A conflicting with a under the given protocol interference model; and $N[a]$ denotes $\{a\} \cup N(a)$. In other words, $N(a)$ (respectively, $N[a]$) is the open (respectively, closed) neighborhood of a in G.
- For any real-valued function f on A and any $B \subseteq A$, $f(B)$ represents $\sum_{b \in B} f(b)$.
- Let $\prec$ be an ordering of A. For any pair of links $a, b \in A$, both $a \prec b$ and $b \succ a$ represent that a appears before b in the ordering $\prec$; $a \preceq b$ represents that either $a \prec b$ or $a = b$; $a \succeq b$ represents that either $a \succ b$ or $a = b$. For any $a \in A$ and any $B \subseteq A$, we use $B_{\prec a}$ (respectively, $B_{\preceq a}$, $B_{\succ a}$, $B_{\succeq a}$) to denote the set of links $b \in B$ satisfying that $b \prec a$ (respectively, $b \preceq a$, $b \succ a$, $b \succeq a$).

II. Orientation-Based Approximation Algorithm

An *orientation* on A is a digraph D obtained from the link-conflict graph G by imposing a direction on each edge of G. Suppose that D is an orientation. For each $a \in A$, $N_D^{in}(a)$ denotes the set of in-neighbors of a in D, and $N_D^{in}[a]$ denotes $\{a\} \cup N_D^{in}(a)$; $N_D^{out}(a)$ denotes the set of out-neighbors of a in D, and $N_D^{out}[a]$ denotes $\{a\} \cup N_D^{out}(a)$. The *inward local independence number* (ILIN) of D is defined to be

$$\alpha_D^{in} = \max_{a \in A} \max_{I \in \mathcal{I}} \left| I \cap N_D^{in}[a] \right|.$$

In this section, we develop a simple approximation algorithm **MWISL** which adopts an orientation D and achieves an approximation ratio nearly $2\alpha_D^{in}$. Conceptually, our algorithm is simple: it first prunes some links from A, and then greedily select an IS from the remaining links in some order. Technically, our algorithm is however intricate. The pruning process is assisted by some auxiliary non-negative weight function x on A. For any $B \subseteq A$, a link $a \in B$ is said to be a *x-surplus link* of B if

$$x\left(N_D^{in}(a) \cap B\right) \geq x\left(N_D^{out}(a) \cap B\right).$$

It was proved in [5] that any non-empty subset B of A has at least one x-surplus link. This fact is exploited by the pruning process repeatedly. After the pruning process, the order in which the IS is greedily selected is also essential. For better exposing our algorithm, this section proceeds as follows. In Subsection II-A, we present the computation of an IS with a given auxiliary weight function. In Subsection II-B, we give a proper selection of the auxiliary weight function. In Subsection II-C, we develop and analyze our algorithm by putting all pieces together and apply it to the specific variants of the protocol interference model.

A. Computing An IS with Auxiliary Weight Function

Suppose that x is an auxiliary non-negative on A with $x(A) > 0$. The algorithm **PG**(x), outlined in Table I, computes an IS I of A with the assistance of x. It maintains a stack S, which is initially empty, and consists of two phases: **Prune Phase** and **Grow Phase**. The **Prune Phase** maintains a set B of links which are yet to be decided whether to be kept (in S) or to be pruned away. Each iteration of the **Prune Phase** selects a x-surplus link a of B and computes an updated weight $\overline{w}(a)$ to be its original weight minus the updated total weight of its conflicting links sitting in the current stack. If the updated weight of a is positive, a is pushed onto the stack S with its updated weight; otherwise, it is pruned from further consideration. Such iteration is repeated until B is empty. The **Grow Phase** builds an IS I of S *incrementally* in the *top-down* manner.

Figure 1 illustrates the **Prune Phase**. Suppose that for the given x, the links are processed in the ordering

$$a_2, a_7, a_6, a_3, a_1, a_5, a_4.$$

The white circles represents the links which have not been processed yet, and the numbers inside the white circles are the original w-weights of the corresponding links. The blue circles

Algorithm $\mathbf{PG}(x)$
//Prune Phase
$B \leftarrow A$, $S \leftarrow \emptyset$; //S is a stack
while $B \neq \emptyset$,
$a \leftarrow$ a x-surplus link of B;
$B \leftarrow B \setminus \{a\}$;
$\overline{w}(a) \leftarrow w(a) - \overline{w}(S \cap N(a))$;
if $\overline{w}(a) > 0$, push a onto S;
//Grow Phase
$I \leftarrow \emptyset$;
while $S \neq \emptyset$,
pop the top link a from S;
if $I \cup \{a\}$ is independent, $I \leftarrow I \cup \{a\}$;
return I.

TABLE I
OUTLINE OF THE ALGORITHM **OrderWIS**.

represents the links which are kept in the current stack, and the numbers inside the blue circles are the updated $\overline{w}$-weights of the corresponding links. The link a_1 is pruned in Figure 1(e) and disappears from then on. At the end of the **Prune Phase**, the stack S consists of six links

$$a_4, a_5, a_3, a_6, a_7, a_2$$

from the top to the bottom whose updated weights are

$$3, 3, 2, 3, 1, 2$$

respectively. During the **Grow Phase**, the vertices a_4, a_6, a_2 are added to the independent set I sequentially.

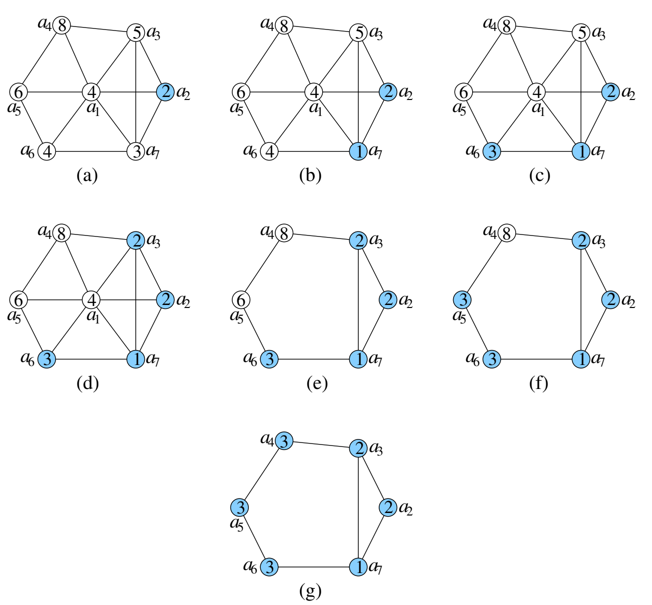

Fig. 1. A step-by-step illustration of the **Prune Phase** in the ordering $a_2, a_7, a_6, a_3, a_1, a_5, a_4$.

Now, we analyze the performance of the algorithm $\mathbf{PG}(x)$.

Theorem 2.1: Let I be the output of the algorithm $\mathbf{PG}(x)$. Then,

$$w(I) \geq \frac{\sum_{a \in A} w(a)\, x(a)}{2 \max_{a \in A} x\left(N_D^{in}[a]\right)}.$$

Proof: Let $\prec$ denote the ordering of A in which the links are processed in the **Prune Phase**. Then, just before a is removed from B, the set B is $A_{\succeq a}$ and so a is a x-surplus link of $A_{\succeq a}$. Thus,

$$\begin{aligned} & x\left(A_{\succeq a} \cap N[a]\right) \\ & = x(a) + x\left(A_{\succeq a} \cap N_D^{in}(a)\right) + x\left(A_{\succeq u} \cap N_D^{out}(a)\right) \\ & \leq x(a) + 2x\left(A_{\succeq a} \cap N_D^{in}(a)\right) \\ & \leq x(a) + 2x\left(N_D^{in}(a)\right) \\ & \leq 2x\left(N_D^{in}[a]\right). \end{aligned}$$

Let S be the set of links in the stack at the end of the **Prune Phase**. Clearly, for each $a \in S$,

$$w(a) = \overline{w}(a) + \overline{w}\left(S_{\prec a} \cap N(a)\right) = \overline{w}\left(S_{\preceq a} \cap N[a]\right);$$

in general, for each $a \in A$,

$$w(a) = \overline{w}(a) + \overline{w}\left(S_{\prec a} \cap N(a)\right) \leq \overline{w}\left(S_{\preceq a} \cap N[a]\right).$$

Now, we claim that

$$\frac{\sum_{a \in A} w(a)\, x(a)}{2 \max_{a \in A} x\left(N_D^{in}(a)\right)} \leq \overline{w}(S)$$

Indeed,

$$\begin{aligned} & \sum_{a \in A} w(a)\, x(a) \\ & \leq \sum_{v \in A} x(a)\, \overline{w}\left(S_{\preceq a} \cap N[a]\right) \\ & = \sum_{b \in S} \overline{w}(b)\, x\left(A_{\succeq b} \cap N[b]\right) \\ & \leq 2 \sum_{b \in S} \overline{w}(b)\, x\left(N_D^{in}[b]\right) \\ & \leq 2\left(\max_{b \in S} x\left(N_D^{in}[b]\right)\right) \sum_{b \in S} \overline{w}(b) \\ & \leq 2\left(\max_{a \in A} x\left(N_D^{in}(a)\right)\right) \overline{w}(S). \end{aligned}$$

So our claim holds.

Next, we claim that

$$w(I) \geq \overline{w}(S).$$

Indeed, by the greedy selection of I, for each $b \in S$,

$$\left|I_{\succeq b} \cap N[b]\right| \geq 1.$$

Thus,

$$\begin{aligned} w(I) & = \sum_{a \in I} \overline{w}\left(S_{\preceq v} \cap N[a]\right) \\ & = \sum_{b \in S} \overline{w}(b) \left|I_{\succeq b} \cap N[b]\right| \\ & \geq \sum_{b \in S} \overline{w}(b) \\ & = \overline{w}(S). \end{aligned}$$

So, our claim holds.

Finally, the lemma holds immediately from the above two claims. ∎

B. Selection of The Auxiliary Weight Function

Theorem 2.1 suggests that the auxiliary weight function x should be selected to make

$$\frac{\sum_{a\in A} w(a)\,x(a)}{\max_{a\in A} x\left(N_D^{in}[a]\right)}$$

as large as possible. Let O be a maximum-weighted independent set O. If x is the indicator function of O, then

$$\frac{\sum_{a\in A} w(a)\,x(a)}{\max_{a\in A} x\left(N_D^{in}[a]\right)} \geq \frac{w(O)}{\alpha_D^{in}}.$$

An auxiliary weight x with maximum

$$\frac{\sum_{a\in A} w(a)\,x(a)}{\max_{a\in A} x\left(N_D^{in}[a]\right)}$$

can actually be obtained by solving the following linear program of linear size:

$$\begin{array}{ll} \max & \sum_{a\in A} w(a)\,x(a) \\ s.t. & x\left(N_D^{in}[a]\right) \leq 1, ,\forall a \in A \\ & x(a) \geq 0, \forall a \in A \end{array}$$

For such x, we have

$$\frac{\sum_{a\in A} w(a)\,x(a)}{\max_{a\in A} x\left(N_D^{in}[a]\right)} \geq \frac{w(O)}{\alpha_D^{in}},$$

and hence the algorithm **PG**(x) produces a $2\alpha_D^{in}$-approximate solution. However, solving the above linear program exactly is time-consuming. In this subsection, we present an efficient price-directive algorithm **PDA**(ε), which takes a parameter $\varepsilon \in (0,1)$ and outputs an auxiliary weight function x satisfying that

$$\frac{\sum_{a\in A} w(a)\,x(a)}{\max_{a\in A} x\left(N_D^{in}[a]\right)} \geq \frac{w(O)}{(1+\varepsilon)\,\alpha_D^{in}}.$$

The running time of **PDA**(ε) increases with $1/\varepsilon$ in at most the square order.

The algorithm **PDA**(ε) uses a notion of price. For any positive function y on A and any $a \in A$, the *y-price* of a is defined to be

$$\frac{y\left(N_D^{out}[a]\right)}{w(a)}.$$

The lemma below shows that the least y-price is no more than $\alpha_D^{in}\frac{y(A)}{w(O)}$.

Lemma 2.2: For any positive function y on A,

$$\min_{a\in A}\frac{y\left(N_D^{out}[a]\right)}{w(a)} \leq \alpha_D^{in}\frac{y(A)}{w(O)}.$$

Proof: On one hand,

$$\begin{aligned} &\sum_{a\in O} y\left(N_D^{out}[a]\right) \\ &= \sum_{a\in O} w(a)\,\frac{y\left(N_D^{out}[a]\right)}{w(a)} \\ &\geq \left(\min_{a\in O}\frac{y\left(N_D^{out}[a]\right)}{w(a)}\right)\sum_{a\in O} w(a) \\ &= \left(\min_{a\in A}\frac{y\left(N_D^{out}[a]\right)}{w(a)}\right) w(O). \end{aligned}$$

On the other hand,

$$\begin{aligned} &\sum_{a\in O} y\left(N_D^{out}[a]\right) \\ &= \sum_{b\in A} y(b)\left|N_D^{in}[b]\cap O\right| \\ &\leq \left(\max_{b\in A}\left|N_D^{in}[b]\cap O\right|\right)\sum_{b\in A} y(b) \\ &\leq \alpha_D^{in}\, y(A). \end{aligned}$$

Thus, the lemma holds. ■

The algorithm **PDA**(ε) is outlined in Table II. Initially, $x(a) = 0$ and $y(a) = 1$ for each $a \in A$; the parameter τ is 0. In each iteration, a least y-priced link a is selected, $x(a)$ is incremented by one, and τ is increased by $\frac{y\left(N_D^{out}[a]\right)}{y(A)}$. Then, for each $b \in N_D^{out}[a]$, $y(b)$ is increased by a factor $1+\varepsilon$. Such iteration is repeated until

$$\max_{a\in A} x\left(N_D^{in}[a]\right) < (1+\varepsilon)\,\tau.$$

Note that the relation

$$y(a) = (1+\varepsilon)^{x\left(N_D^{in}[a]\right)}, \forall a \in A$$

holds at the initialization and is maintained at the end of each iteration.

Algorithm **PDA**(ε)
$\forall a \in A, x(a) \leftarrow 0, y(a) \leftarrow 1; \tau \leftarrow 0;$
while $\max_{a\in A} x\left(N_D^{in}[a]\right) \geq (1+\varepsilon)\,\tau$ do
$\quad a \leftarrow \arg\min_{a\in A}\frac{y\left(N_D^{out}[a]\right)}{w(a)};$
$\quad \tau \leftarrow \tau + \frac{y\left(N_D^{out}[a]\right)}{y(A)};$
$\quad x(a) \leftarrow x(a) + 1;$
$\quad \forall b \in N_D^{out}[a], y(b) \leftarrow (1+\varepsilon)\,y(b);$
return x.

TABLE II
OUTLINE OF THE ALGORITHM **PDA**.

Next, we analyze the performance of the algorithm **PDA**(ε).

Theorem 2.3: The algorithm **PDA**(ε) terminates in at most

$$\left\lceil \frac{|A|\ln|A|}{\ln(1+\varepsilon) - \frac{\varepsilon}{1+\varepsilon}} \right\rceil$$

iterations, and outputs x satisfying that

$$\frac{\sum_{a\in A} w(a)\, x(a)}{\max_{a\in A} x\left(N_D^{in}[a]\right)} \geq \frac{w(O)}{(1+\varepsilon)\,\alpha_D^{in}}.$$

Proof: We introduce the following notations in this proof. Let $m = |A|$. x_0, y_0, and τ_0 denote initial values of x, y, and τ respectively. For each iteration $i \geq 1$, x_i, y_i and τ_i denote the values of x, y, and τ respectively at the end of the i-th iteration; a_i denotes the link selected in the i-th iteration. Then, in each iteration i,

$$\tau_i - \tau_{i-1} = \frac{y_{i-1}\left(N_D^{out}[a_i]\right)}{y_{i-1}(A)},$$

and

$$\begin{aligned} y_i(A) &= y_{i-1}(A) + \varepsilon y_{i-1}\left(N_D^{out}[a_i]\right) \\ &= y_{i-1}(A)\left(1 + \varepsilon(\tau_i - \tau_{i-1})\right) \\ &\leq y_{i-1}(A)\exp\left(\varepsilon(\tau_i - \tau_{i-1})\right) \end{aligned}$$

We claim that for any iteration number k, at the end of the k-th iteration,

$$\frac{1}{\varepsilon}\ln\frac{y_k(A)}{m} \leq \tau_k \leq \frac{\alpha_D^{in}\sum_{a\in A} w(a)\, x_k(a)}{w(O)}$$

Indeed, by Lemma 2.2,

$$\begin{aligned} \tau_k &= \sum_{i=1}^{k}\frac{y_{i-1}\left(N_D^{out}[a_i]\right)}{y_{i-1}(A)} \\ &\leq \frac{\alpha_D^{in}\sum_{i=1}^{k} w(a_i)}{w(O)} \\ &= \frac{\alpha_D^{in}\sum_{a\in A} w(a)\, x_k(a)}{w(O)}. \end{aligned}$$

By induction,

$$\begin{aligned} y_k(A) &\leq y_0(A)\exp\left(\varepsilon\sum_{t=1}^{k}(\tau_i - \tau_{i-1})\right) \\ &= m\exp(\varepsilon\tau_k), \end{aligned}$$

which implies

$$\tau_k \geq \frac{1}{\varepsilon}\ln\frac{y_k(A)}{m}.$$

Thus, our claim holds.

Now, we bound the number of iterations of the algorithm. Assume to the contrary that the algorithm didn't terminate after

$$\left\lceil\frac{m\ln m}{\ln(1+\varepsilon) - \frac{\varepsilon}{1+\varepsilon}}\right\rceil$$

iterations. Let

$$k = \left\lceil\frac{m\ln m}{\ln(1+\varepsilon) - \frac{\varepsilon}{1+\varepsilon}}\right\rceil.$$

Since $\sum_{a\in A} x_i\left(N_D^{in}[a]\right)$ strictly increases with the iteration i,

$$\sum_{a\in A} x_k\left(N_D^{in}[a]\right) \geq k.$$

Let a be the link in A maximizing $x_k\left(N_D^{in}[a]\right)$. Then,

$$x_k\left(N_D^{in}[a]\right) \geq k/m \geq \frac{\ln m}{\ln(1+\varepsilon) - \frac{\varepsilon}{1+\varepsilon}}.$$

Hence,

$$\frac{1}{\varepsilon}\ln\frac{(1+\varepsilon)^{x_k\left(N_D^{in}[a]\right)}}{m} \geq \frac{x_k\left(N_D^{in}[a]\right)}{1+\varepsilon}.$$

Thus,

$$\begin{aligned} \tau_k &\geq \frac{1}{\varepsilon}\ln\frac{y_k(A)}{m} \\ &> \frac{1}{\varepsilon}\ln\frac{(1+\varepsilon)^{x_k\left(N_D^{in}[a]\right)}}{m} \\ &\geq \frac{x_k\left(N_D^{in}[a]\right)}{1+\varepsilon}. \end{aligned}$$

This means that the number of iterations is at most k, which is a contradiction.

Finally, we show the output x satisfies the inequality stated in the theorem. Suppose that the algorithm runs in k iterations. By the stopping rule of the algorithm,

$$\begin{aligned} &\max_{a\in A} x_k\left(N_D^{in}[a]\right) \\ &\leq (1+\varepsilon)\,\tau_k \\ &\leq \frac{(1+\varepsilon)\,\alpha_D^{in}\sum_{a\in A} w(a)\, x_k(a)}{w(O)}. \end{aligned}$$

Thus,

$$\frac{\sum_{a\in A} w(a)\, x_k(a)}{\max_{a\in A} x_k\left(N_D^{in}[a]\right)} \geq \frac{w(O)}{(1+\varepsilon)\,\alpha_D^{in}}.$$

This completes the proof of the theorem. ■

As

$$\begin{aligned} \ln(1+\varepsilon) &= -\ln\left(1 - \frac{\varepsilon}{1+\varepsilon}\right) \\ &\geq \frac{\varepsilon}{1+\varepsilon} + \frac{1}{2}\left(\frac{\varepsilon}{1+\varepsilon}\right)^2, \end{aligned}$$

we have

$$\begin{aligned} \ln(1+\varepsilon) - \frac{\varepsilon}{1+\varepsilon} &\geq \frac{1}{2}\left(\frac{\varepsilon}{1+\varepsilon}\right)^2 \\ &= \frac{1}{2}(1 + 1/\varepsilon)^{-2}. \end{aligned}$$

The running time of **PDA**(ε) increases with $1/\varepsilon$ in at most the square order.

C. Putting Together

Now, we are ready describe our orientation-based approximation algorithm **OrientWIS**(ε) for **MWISL**. The algorithm takes a parameter $\varepsilon \in (0,1)$ and produces an IS I of A in two steps:

- Apply the algorithm **PDA**(ε) to compute an auxiliary weight function x on A.

- Apply the algorithm $\mathbf{PG}(x)$ to compute an IS I of A.

From Theorem 2.1 and Theorem 2.3, we immediately obtain the following approximation bound of the algorithm **OrientWIS**(ε).

Theorem 2.4: The approximation ratio of **OrientWIS**(ε) is $2(1+\varepsilon)\alpha_D^{in}$.

Next, we apply the algorithm **OrientWIS**(ε) to the plane geometric variants of the protocol interference model. In the bidirectional mode, Wan et al. [6] introduced the following orientation D. Consider any conflicting pair of links a and b in A. If a has an endpoint u and b has an endpoint v satisfying that u is within the interference range of v and its interference radius is no more than that of v, then we take the orientation from b to a; otherwise, we take the orientation from a to b. Ties are broken arbitrarily. It was shown in [6] that the ILIN of such D is at most 8. By Theorem 2.4, we have the following corollary.

Corollary 2.5: Under the plane geometric variant of the protocol interference model in the bidirectional mode, by adopting the orientation D given in [6], the algorithm **OrientWIS**(ε) has an approximation bound $16(1+\varepsilon)$.

In the unidirectional mode, Wan [5] introduced the following orientation D. Consider any conflicting pair of links a and b in A. If the receiver of a is within the interference range of the sender of b, then we take the orientation from b to a; otherwise, we take the orientation from a to b. Ties are broken arbitrarily. Suppose that for each link $a \in A$, the interference radius of its sender is at least c times its length for some constant $c > 1$. It was shown in [6] that the ILIN of such D is at most

$$\left\lceil \pi / \arcsin \frac{c-1}{2c} \right\rceil - 1.$$

By Theorem 2.4, we have the following corollary.

Corollary 2.6: Under the plane geometric variant of the protocol interference model in the unidirectional mode in which the interference radius of its sender of each link is at least c times the link length for some constant $c > 1$, by adopting the orientation D given in [5], the algorithm **OrientWIS**(ε) has an approximation bound

$$2(1+\varepsilon)\left(\left\lceil \pi / \arcsin \frac{c-1}{2c} \right\rceil - 1\right),$$

III. Ordering-Based Approximation Algorithm

For any ordering $\prec$ of A, the *forward local independence number* (FLIN) of $\prec$ is defined to be

$$\alpha^{\prec} = \max_{a \in A} \max_{I \in \mathcal{I}} |I_{\succeq a} \cap N[a]|.$$

An ordering $\prec$ naturally defines an orientation on D as follows: For any conflicting pair of links a and b in A, if $a \prec b$, we take the orientation from b to a; otherwise, we take the orientation from a to b. For such orientation D, its ILIN is exactly $\alpha^{\prec}$. By adopting such orientation D, the algorithm **OrientWIS**(ε) has an approximation bound $2(1+\varepsilon)\alpha^{\prec}$. However, an ordering is stronger than a general orientation in the sense that the orientation defined by an ordering is acyclic. In this section, we present an ordering based approximation **OrderWIS** which takes advantage of such stronger property of orderings. The algorithm **OrderWIS** is not only simpler, but also achieves an approximation bound $\alpha^{\prec}$, where $\prec$ is the ordering adopted by the algorithm.

The algorithm **OrderWIS** is outlined in Table III. It adopts an ordering $\prec$ of A and is a simplified adaptation from the algorithm **OrientWIS**(ε). It maintains a stack S and consists of two phases: **Prune Phase** and **Grow Phase**. In the **Prune Phase**, for each link $a \in A$ in the ordering $\prec$, it computes its updated weight $\overline{w}(a)$ to be its original weight minus the updated total weight of its conflicting links sitting in the current stack. If the updated weight of a is positive, a is pushed onto the stack S with its updated weight; otherwise, it is pruned from further consideration. The **Grow Phase** builds an IS I of S incrementally in the top-down manner. The performance of the algorithm **OrderWIS** is given in the theorem below.

Algorithm **OrderWIS**
//Prune Phase
$S \leftarrow \emptyset$, $I \leftarrow \emptyset$;
for each $a \in A$ in the given ordering $\prec$
$\overline{w}(a) \leftarrow w(a) - \overline{w}(S \cap N(a))$;
if $\overline{w}(a) > 0$, push a onto S;
//Grow Phase
$I \leftarrow \emptyset$;
while $S \neq \emptyset$
pop the top link a from S;
if $I \cup \{a\}$ is independent, $I \leftarrow I \cup \{a\}$;
return I.

TABLE III
OUTLINE OF THE ALGORITHM **OrderWIS**.

Theorem 3.1: The approximation ratio of **OrderWIS** is at most $\alpha^{\prec}$.

Proof: Let S be the set of links in the stack at the end of the **Prune Phase**. Clearly, for each $a \in S$,

$$w(a) = \overline{w}(S_{\preceq a} \cap N[a]);$$

in general, for each $a \in A$,

$$w(a) \leq \overline{w}(S_{\preceq a} \cap N[a]).$$

Let O be an optimal solution. Then,

$$\begin{aligned} w(O) &\leq \sum_{a \in O} \overline{w}(S_{\preceq a} \cap N[a]) \\ &= \sum_{b \in S} \overline{w}(b) |O_{\succeq b} \cap N[b]| \\ &\leq \left(\max_{b \in S} |O_{\succeq b} \cap N[b]|\right) \sum_{b \in S} \overline{w}(b) \\ &\leq \alpha^{\prec} \overline{w}(S). \end{aligned}$$

Let I be the output by the algorithm. By using the same argument in the proof of Theorem 2.1, we can show that

$$w(I) \geq \overline{w}(S).$$

Therefore,

$$w(O) \leq \alpha^{\prec}\overline{w}(S) \leq \alpha^{\prec}w(I).$$

So, the theorem holds. ■

Next, we apply the algorithm **OrderWIS** to the plane geometric variants of the protocol interference model in the bidirectional mode. We define the interference radius of a link to be the larger one between the interference radii of its two endpoints. The interference-radius increasing ordering sorts the links in the increasing order of their interference radii and ties are broken arbitrarily. For arbitrary interference radii, its FLIN is at most 23 [5]. For symmetric interference radii (i.e, for each link, its two endpoints have equal interference radii), its FLIN is at most 8 [6]. By Theorem 3.1, we have the following corollary.

Corollary 3.2: Under the plane geometric variant of the protocol interference model in the bidirectional mode, by adopting the interference radius increasing ordering, the approximation ratio of **OrderWIS** is at most 23 for arbitrary interference radii, and at most 8 for symmetric interference radii.

We remark that for arbitrary interference radii, the 23-approximation bound of **OrderWIS** is larger than the $16(1+\varepsilon)$-approximation bound of **OrientWIS**(ε). However, **OrderWIS** enjoys a simpler implementation than **OrderWIS.**

In case of uniform interference radii, we consider a different ordering. The lexicographic ordering sorts the links in the lexicographic order of their left endpoints and ties are broken arbitrarily. The *reverse* of the lexicographic ordering has FLIN at most 6 [3]. By Theorem 3.1, we have the following corollary.

Corollary 3.3: Under the plane geometric variant of the protocol interference model in the bidirectional mode with uniform interference radii, by adopting the reverse of the lexicographic ordering, the approximation ratio of **OrderWIS** is at most 6.

We remark that for uniform interference radii, the 6-approximation bound of **OrderWIS** is better than the 8-approximation bound of the simple greedy algorithm in [1].

IV. Divide And Conquer

Under the plane geometric variants of the protocol interference model with uniform interference radii, we further present a better approximation algorithm for **MWISL** which exploits the following strip-wise transitivity of independence discovered in [7]. Suppose that the maximum link length is normalized to one, and all nodes have an interference radius $r \geq 1$. In the unidirectional mode, we assume that $r > 1$.

- Bidirectional mode: Let S be a horizontal strip of height
$$h(r) = \sqrt{r^2 - \frac{1}{4}}\cos\left(\frac{\pi}{6} + \arcsin\frac{1}{2r}\right). \tag{1}$$
Suppose that a_1, a_2 and a_3 are three links whose *mid-points* lie in S from left to right (see Figure 2). If both a_1 and a_3 are independent with a_2, then a_1 and a_3 are also independent with each other.
- Unidirectional mode: Let S be a horizontal strip of height
$$h(r) = (r-1)\sin\left(\arccos\frac{r-1}{2r} - \arcsin\frac{1}{r}\right). \tag{2}$$
Suppose that a_1, a_2 and a_3 are three links whose *senders* lie in S from left to right (see Figure 3). If both a_1 and a_3 are independent with a_2, then a_1 and a_3 are also independent with each other.

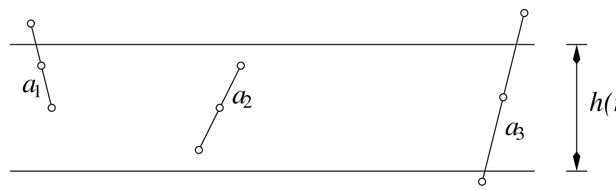

Fig. 2. Strip-wise transtivity of independence in bidirectional mode.

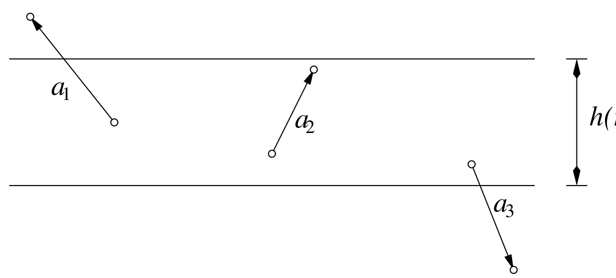

Fig. 3. Strip-wise transtivity of independence in unidirectional mode.

In Subsection IV-A, we give a greedy algorithm **GWIS** for **MWISL** restricted to the sets of links satisfying the transitivity of independence in some ordering. In Subsection IV-B, we present a spatial divide-and-conquer approximation algorithm **StripWIS** for **MWISL** under the plane geometric variants of the protocol interference model with uniform interference radii, which utilizes the algorithm **GWIS** to solve the sub-problems.

A. A Greedy Algorithm

Suppose that B is a subset of A which has an ordering $\prec$ satisfying the transitivity of independence, i.e., for any three links a_1, a_2 and a_3 in B with $a_1 \prec a_2 \prec a_3$, the independence of $\{a_1, a_2\}$ and $\{a_2, a_3\}$ implies the independence of $\{a_1, a_3\}$. A maximum-weighted IS of B can be computed in polynomial time by a greedy algorithm **GWIS** given in this subsection.

We first describe the underlying recursive relation of the algorithm **GWIS**. Consider any $a \in B$. Let $\mathcal{I}^{\prec}[a]$ denote the collection of independent subsets of $B_{\preceq a}$ which contains a itself, and $f(a)$ be the weight of a maximum-weighted independent set in $\mathcal{I}^{\prec}[a]$. A link $b \in B$ is said to be an *immediate independent predecessor* of a if (1) $b \prec a$, (2) $\{a, b\} \in \mathcal{I}$, and (3) for any link $b' \in B$ with $b \prec b' \prec a$, $\{a, b, b'\} \notin \mathcal{I}$. Let $J(a)$ denote the set of all immediate independent predecessors of a. Clearly, if $J(a) = \emptyset$, then $\{a\}$ is the only independent set in $\mathcal{I}^{\prec}[a]$ and $f(a) = w(a)$. If $J(a) \neq \emptyset$, then, we have the following recursive relation.

Lemma 4.1: Consider any $a \in B$ with $J(a) \neq \emptyset$. Let b any link in $J(a)$ with maximum $f(b)$ and I be any set in $\mathcal{I}^{\prec}[b]$ with maximum weight. Then, $\{a\} \cup I$ is a maximum weighted set in $\mathcal{I}^{\prec}[a]$; and consequently,

$$f(a) = w(a) + \max_{b \in J(a)} f(b);$$

Proof: By the transitivity of independence, $\{a\} \cup I \in \mathcal{I}^{\prec}[a]$. We prove by contradiction that $w(\{a\} \cup I) = f(a)$. Assume to the contrary that $w(\{a\} \cup I) < f(a)$. Let $I' \in \mathcal{I}^{\prec}[a]$ be such that $w(I') = f(a)$. Then, $I' \setminus \{a\} \neq \emptyset$. Let b' be the last link in $I' \setminus \{a\}$ in the ordering $\prec$. Then, $I' \setminus \{a\} \in \mathcal{I}^{\prec}[b']$, and hence

$$\begin{aligned} f(b') &\geq w(I' \setminus \{a\}) \\ &= f(a) - w(a) \\ &> w(\{a\} \cup I) - w(a) \\ &= w(I) \\ &= f(b). \end{aligned}$$

By the choice of b, $b' \notin J(a)$. So, there must exist a link $b'' \in B$ satisfying that $b' \prec b'' \prec a$ and $\{a, b', b''\} \in \mathcal{I}$. Again by transitivity of independence, $I' \cup \{b''\}$ is independent, and hence $I' \cup \{b''\} \in \mathcal{I}^{\prec}[a]$. But

$$w(I' \cup \{b''\}) > w(I') = f(a),$$

which is a contradiction. Thus, we must have that $w(\{a\} \cup I) = f(a)$. So, the lemma holds. ■

The algorithm **GWIS** is outlined in Table IV. For the reconstruction of a maximum weighted independent set I, a variable $pre(a)$ of each link a is defined as follows. If $J(a) = \emptyset$ is empty, then $pre(a)$ is null; otherwise, $pre(a)$ is some link $b \in J(a)$ with maximum $f(b)$. Then, both $f(a)$ and $pre(a)$ can be computed sequentially in the order $\prec$ using the recursive relation proved in Lemma 4.1. After the completion of the computations of $f(a)$ and $pre(a)$ for all links $a \in B$, a maximum-weighted IS I of B is reconstructed as follows. Let a be the link in B with maximum $f(a)$, and initially I consists of a only. While $pre(a)$ is not null, $pre(a)$ is added to I and a is reset to $pre(a)$. The final I is a maximum-weighted IS of B.

B. Divide And Conquer

The algorithm **StripWIS** takes a divide-and-conquer approach. We describe its three algorithmic components division, conquer, and combination below.

Division: The same division of A given in [7] is adopted here. A *representative* of a link is defined to be its midpoint

Algorithm **GWIS**

for each $a \in B$ in $\prec$ do
    compute $J(a)$;
    if $J(a) = \emptyset$ then
        $pre(a) \leftarrow null$; $f(a) \leftarrow w(a)$;
    else
        $pre(a) \leftarrow \arg\max_{b \in J(a)} f(b)$;
        $f(a) \leftarrow w(a) + f(pre(a))$;

$a \leftarrow \arg\max_{a \in B} f(a)$; $I \leftarrow \{a\}$; while $pre(a) \neq null$ do $I \leftarrow I \cup \{pre(a)\}$;$a \leftarrow pre(a)$; return I.

TABLE IV
OUTLINE OF THE ALGORITHM **GWIS**.

(respectively, sender) in the bidirectional (respectively, unidirectional) mode. Let

$$\mu = \left\lceil \frac{r+1}{h(r)} \right\rceil + 1,$$

where $h(r)$ is given by Equation (1) (respectively, Equation (2)) in the bidirectional (respectively, unidirectional) mode. The minimal axis-parallel rectangle surrounding the representatives of all links in A is computed and partitioned into top-closed bottom-open horizontal strips in the manner that the upper boundary of the top-most strip aligns with the top of the rectangle, the heights of all strips except the bottom-most one are all equal to $(r+1)/(\mu-1)$, and the height of the bottom-most strip is at most $(r+1)/(\mu-1)$ (see Figure 4). Let l denote the total number of strips, and number the successive strips from top to bottom using integers $0, 1, \cdots, l-1$. For each $0 \leq i \leq l-1$, let A_i be the set of links in A whose representatives lie in the i-th strip.

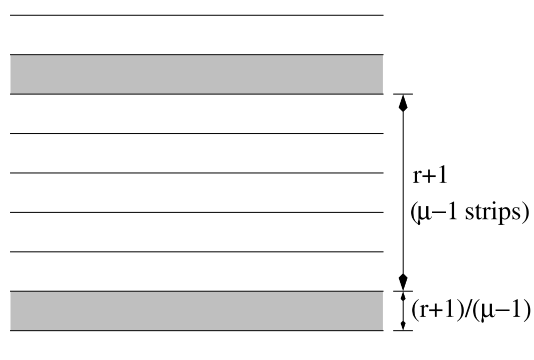

Fig. 4. Partition of the plane into half-open half-closed strips of height $(r+1)/(\mu-1)$ where $\mu = \lceil (r+1)/h(r) \rceil + 1$.

Conquer: Since the height of each strip is at most

$$(r+1)/(\mu-1) \leq h(r),$$

each A_i satisfies the transitivity of independence in the lexicographic ordering of the representatives of A_i. So, the algorithm **GWIS** is applied to compute a maximum-weighted independent set M_i of A_i for each $0 \leq i \leq l-1$.

Combination: For each $0 \leq j < \mu$, let I_j be the union of the sets M_i with $0 \leq i < l$ and $i = j \bmod \mu$. Then, each I_j is still independent as argued in [7]. Among these μ independent sets I_j for $0 \leq j < \mu$, the one with maximum weight is output by the algorithm **StripWIS**.

Theorem 4.2: The approximation ratio of the algorithm **StripWIS** is at most μ.

Proof: Let O be a maximum weighted IS of A. For each $0 \leq i \leq l-1$, let $O_i = O \cap A_i$. Then, $w(O_i) \leq w(M_i)$ for each $0 \leq i \leq l-1$. Hence,

$$\begin{aligned} w(O) &= \sum_{i=0}^{l-1} w(O_i) \leq \sum_{i=0}^{l-1} w(M_i) = \sum_{j=0}^{\mu-1} w(I_j) \\ &\leq \mu \max_{0 \leq j \leq \mu-1} w(I_j). \end{aligned}$$

So, the theorem holds. ∎

The value of μ was computed in [7]. In the bidirectional mode,

$$\mu = \begin{cases} 6 & \text{if } r \in [1, 1.0891); \\ 5 & \text{if } r \in [1.0891, 1.3609); \\ 4 & \text{if } r \in [1.3609, 2.2907); \\ 3 & \text{if } r \in [2.2907, \infty). \end{cases}$$

In the unidirectional mode, $\mu = k+1$ over $[r_k, r_{k-1})$ for any $k \geq 2$, where r_k is the unique root of the following quartic polynomial in $(1, \infty)$:

$$\begin{aligned} &\left(4-3k^2\right) r^4 + 4\left(k^2+k+2\right) r^3 + 2\left(3k^2-2k+2\right) r^2 \\ &- 4k(3k+1) r + \left(5k^2+4k\right). \end{aligned}$$

The numeric values of r_k can be computed with the quartic formula. Table V lists the numeric values of r_k for $2 \leq k \leq 11$.

k	r_k	k	r_k
2	4.2462	7	1.5715
3	2.5689	8	1.5009
4	2.0632	9	1.4476
5	1.8167	10	1.4058
6	1.6697	11	1.3721

TABLE V
NUMERIC VALUES OF r_k FOR $2 \leq k \leq 11$.

V. CONCLUSION

In this paper, we have developed several fast and simple approximation algorithms for **MWISL** under the general protocol interference model. These algorithms imply the following impact on the structural properties of protocol interference on the approximality of **MWISL**:

- If there is an orientation of ILIN μ, then for any $\varepsilon > 0$ a simple $2(1+\varepsilon)\mu$-approximate IS can be computed efficiently in polynomial time.
- If there is an ordering of FLIN μ, then a μ-approximate IS can be computed efficiently in polynomial time.
- If there is an ordering satisfying the transitivity of independence, then a maximum-weighted IS can be computed efficiently in polynomial time.

By exploiting the rich nature of the plane geometric variant of the protocol interference model discovered in the literature, these algorithms are able to produce constant-approximate solutions efficiently.

Due to the existence of polynomial approximation-preserving reductions to **MWISL** [5], minimum-latency link scheduling, maximum multiflow, and maximum concurrent multiflow also have the same approximality, which was already known in the literature [5], [7]. However, because of the fractional nature of these three problems (i.e., an independent set can be scheduled for a fractional amount of time), the same approximality can be achieved by simpler approximation algorithms. The integral nature of **MWISL** (i.e., a link is selected or not selected by the output independent set) makes the same approximality harder to achieve in most settings. Indeed, our orientation-based approximation and ordering-based approximation both have to run a subtle pruning process before the standard greedy growing process of building up an independent set.

ACKNOWLEDGEMENTS: This work was supported in part by the National Science Foundation of USA under grants CNS-0916666 and CNS-1219109, by the National Natural Science Foundation of P. R. China under grants 61128005, 61100191 and 61370216, by Natural Scientific Research Innovation Foundation in Harbin Institute of Technology under Project HIT-NSFIR 2011128, and by Shenzhen Strategic Emerging Industries Program under grant ZDSY20120613125016389.

REFERENCES

[1] P. Chaporkar, K. Kar, X. Luo, and S. Sarkar, Throughput and Fairness Guarantees Through. Maximal Scheduling in Wireless Networks, *IEEE Transactions on Information Theory* 54(2):572–594, 2008.

[2] C. Joo, X. Lin, and N. B. Shroff, Greedy Maximal Matching: Performance Limits for Arbitrary Network Graphs Under the Node-exclusive Interference Model, *IEEE Transactions on Automatic Control* 54(12): 2734-2744, 2009.

[3] C. Joo, X. Lin, and N. B. Shroff, Understanding the Capacity Region of the Greedy Maximal Scheduling Algorithm in Multi-hop Wireless Networks, in *Proc. IEEE INFOCOM'08*, April 2008.

[4] X. Lin and N. B. Shroff, The Impact of Imperfect Scheduling on Cross-Layer Rate Control in Wireless Networks, *IEEE/ACM Transactions on Networking* 14(2):302–315, 2006.

[5] P.-J. Wan, Multiflows in Multihop Wireless Networks, *ACM MOBIHOC* 2009, pp. 85-94.

[6] P.-J. Wan, C. Ma, Z. Wang, B. Xu, M. Li, and X. Jia, Weighted Wireless Link Scheduling without Information of Positions And Interference/Communication Radii, in *Proc. IEEE INFOCOM* 2011.

[7] P.-J. Wan, L. Wang, A. Huang, M. Li, and F. Yao, Approximate Capacity Subregions of Uniform Multihop Wireless Networks, *IEEE INFOCOM* 2010.