Diffusion based Distributed Internet Gateway Load Balancing in a Wireless Mesh Network

Bing He
Department of Computer Science
University of Cincinnati
Cincinnati, OH 45221-0030
Email: heb@email.uc.edu

Dongmei Sun
Institute of Information Science
Beijing Jiaotong University
Beijing 100044, China
Email: dmsun@bjtu.edu.cn

Dharma P. Agrawal
Department of Computer Science
University of Cincinnati
Cincinnati, OH 45221-0030
Email: dpa@cs.uc.edu

Abstract—In a wireless mesh network (WMN), mesh clients (MCs) access the Internet through the wireless backbone formed by the Internet Gateways (IGWs) and Mesh Routers (MRs). An IGW as the traffic center may easily become the congestion center in a WMN. The load balancing between IGWs will reduce the traffic congestion thereby improving the network performance and providing a better quality of service (QoS). In this paper, potential load balancing between different IGW service domains is examined and fairness metrics are defined for both homogenous and heterogeneous WMNs. A distributed load balancing scheme based on diffusion methods is proposed to achieve a global load fairness. The effectiveness and efficiency of proposed load balancing scheme are verified by extensive simulations.

I. INTRODUCTION

A Wireless Mesh Network (WMN) basically integrates multi-hop communication with the Internet, and is expected to provide ubiquitous Internet connectivity to a large number of mobile clients (MCs) for various services [1]. In a WMN as shown in Fig.1, a MC can access the Internet through a wireless backbone formed by wireless Mesh Routers (MRs) and Internet Gateways (IGWs). MRs are interconnected in a multi-hop fashion while IGWs are specialized MRs acting as the communication bridges between the wireless backbone and the Internet.

In a WMN, the traffic from the mobile clients is predominantly directed between the MCs and the Internet via some MRs, and all information pass through the IGWs. The throughput for mobile clients is primarily restricted by the IGW and a WMN could get congested at an IGW, even though the individual MRs can support a higher bandwidth. In the popular contention based wireless media access protocols such as CSMA/CA in IEEE 802.11, contention for the channel will be extremely fierce around IGW when the traffic load is heavy. This degrades the network performance, in terms of increased packet delay and higher packet loss probability [2]. At a given time, some IGWs may be heavily loaded while others may have very little traffic. To reduce the congestion around a heavily loaded IGW, a load-balancing approach is desirable to divert its traffic toward an under-utilized IGW. This makes resource utilization uniform throughout the network and the overall network performance is improved. In this paper, we address the load balancing problem at IGWs of WMNs. To determine a global fairness level, load balance metrics are defined for both homogenous and heterogeneous WMNs. A distributed load balancing scheme is introduced to dynamically balance the traffic between different IGWs. Three different diffusion algorithms are investigated. To the best of our knowledge, this is the first work that provides improvement of IGW load fairness in the entire network through distributed load balancing. Our scheme can achieve a better global load balancing without centralized coordination or management, thus possesses good scalability and robustness.

The remainder of this paper is organized as follows. In section II, background and related work are addressed. In section III, a generic WMN network model is defined, and the IGW load balancing problem is formulated. In section IV, evaluation metric for load balancing is discussed, and the fairness indices are defined both for homogenous and heterogeneous WMNs. In section V, a distributed load balancing scheme is proposed and three diffusion based load balancing algorithms are presented. Section VI examines the effectiveness of the proposed load balancing algorithms by extensive simulations. Finally, concluding remarks are included in section VII.

Fig. 1. WMN Network Model
II. BACKGROUND AND RELATED WORK

Load balancing is important for avoiding congestion, increasing network throughput, and providing reliability in case of any failure [2], [3]. Efficient load balancing in WMNs becomes even more challenging due to limitations on MRs’ available bandwidth and contention for wireless links. This can be classified into two categories: path and Internet gateway load balancing. The path load balancing is aimed at improving the network throughput and reliability by diverting traffic to paths with lower load, no traffic conflict and minimum channel interference [4]–[6]. Another important type of load balancing is the Internet gateway load balancing. In a WMN, IGW is always an aggregation point for flows, could become bottleneck due to the finite number of radio interfaces and given channel bandwidth. It is believed that the performance improvement due to load balancing at IGWs will be greater than at the path [2], [7]. In the IGW load balancing, the affiliation between MRs and IGWs are dynamically changed based on the load of a particular IGW. Scheme proposed by Nandiraju et. al. [8] detects the congestion at an IGW by monitoring the average queue length at the IGW, and MRs switch to less congested gateways. The domain partitioning approach proposed in [3] provides load balancing between neighboring domains, along with an Internet mobility approach for handling the MCs inter-domain mobility. Choi and Han proposed a partition based algorithm [9] without considering load balancing at the global level. All these schemes lack any systematic global load balancing approach, and the migration is done intuitively.

III. NETWORK MODEL AND PROBLEM STATEMENT

A WMN can be formulated as a multi-radio, multi-channel, multi-hop, and infrastructure-based wireless network, which can be represented by an undirected graph \( G=(V, E) \), \( V = \{v_1, \ldots, v_n\} \) is the set of \( n \) network nodes, representing MRs and IGWs. The physical location of a node \( v_i \in V \) is static after deployment. Every MR \( v_i \) may be configured with one or multiple wireless interfaces. Among \( n \) MR nodes, there are \( m \) nodes configured as IGWs which provide the wired Internet connection. In a WMN, every MR aggregates traffic from MCs. Unlike a pure ad hoc network where traffic is randomly generated between peer nodes, the traffic of a MR in a WMN is predominantly directed either towards the IGW or from the IGW to the MR. The IGW oriented Internet traffic in WMN in turn implies the IGW-oriented network architecture. At a given time period, each MR is only associated with one IGW, i.e., it accesses Internet through one specific IGW. This is reasonable as it is difficult to maintain associations with different IGWs, especially when MR is equipped with only one radio [6]. Thus, the whole WMN are organized into several IGW domains \( D_1, \ldots, D_m \), and each IGW domain \( D_i \) consists of one IGW \( I_i \) and several associated MRs. For example, the WMN in Fig. 1 consists of four IGW domains, in which four IGWs \( I_1, I_2, I_3, I_4 \) provide Internet access for MRs in each domain respectively. Each IGW domain is usually organized as an IGW rooted spanning tree [7], and each MR can connect to the IGW via one tree path, which is usually limited to hop count \( h_{max} \). The aggregated traffic of MR can be changed dynamically as a result of MC movements. Each IGW has a cumulative load which is the sum of the traffic coming from its affiliated MRs in its domain. Due to the limitation of the available number of radio interfaces and the bandwidth of each channel, each IGW \( I_i \) has a maximum throughput capacity, denoted by \( Cap_i \). We name MR as a “bordering” node if it could be connected to multiple IGWs within allowable hop count (i.e. \( h_{max} \)). Such IGW domains are called “neighboring” IGW domains. For instance, in the WMN shown in Fig. 1 with \( h_{max} = 3 \), \( MR_1 \) is the bordering MR between \( D_1 \) and \( D_2 \), and \( MR_2 \) is the bordering MR between \( D_2 \) and \( D_3 \). We denote \( Neighbor(i) \) as the neighboring domains set of domain \( D_i \).

In this paper, we consider the problem of load balancing among multiple IGW domains, which implies partitioning of a WMN. When an IGW is highly loaded, a load-balancing approach should be able to redirect part of the traffic from an overloaded to a neighboring lightly loaded domain. Such reallocation of traffic flow means an inter-domain migration of associated MRs. After the load adjustment process, the network enters a state with a better load fairness.

Depending on how information are collected for MRs migration among different domains, load balancing can be classified into centralized and distributed approach. Centralized schemes tend to be more accurate because they collect and use the global load information of all the IGW domains in the network. Nevertheless, these approaches require excessive communication overhead and are not easily scalable. On the other hand, in distributed approaches the load information from neighboring domains is used by each IGW in making the load balancing decision.

IV. EVALUATION INDEX OF LOAD BALANCING

First, we define the load balance index \( \beta(m) \) among \( m \) IGW service domains. The load balancing metric is based on the Chebyshev’s sum inequality, which states that if:

\[
a_1 \geq a_2 \geq \cdots \geq a_n, \quad \text{and} \quad b_1 \geq b_2 \geq \cdots \geq b_n,
\]

then

\[
n \sum_{k=1}^{n} a_k b_k \geq (\sum_{k=1}^{n} a_k)(\sum_{k=1}^{n} b_k).
\]

When \( a_k = b_k \), the inequality becomes

\[
n \sum_{k=1}^{n} a_k^2 \geq (\sum_{k=1}^{n} a_k)^2.
\]

(1)

The equality hold only when \( a_1 = a_2 = \cdots = a_n \).

A. Load Balancing Metric in a Homogenous WMN

Based on the Eq. 1, the load fairness index \( \beta(m) \) in a WMN with \( m \) IGWs with the same capacity can be defined as:

\[
\beta(m) = \frac{\left( \sum_{i=1}^{m} Load_i \right)^2}{m \sum_{i=1}^{m} Load_i^2} = \frac{\left( \sum_{i=1}^{m} \sum_{j=1}^{n} load_{i,j} \right)^2}{m \sum_{i=1}^{m} \left( \sum_{j=1}^{n} load_{i,j} \right)^2}.
\]

(2)
where $i$ donates the $i$th domain $D_i$ in a WMN and $Load_i$ is the aggregated traffic load in $D_i$. $load_{i,j}$ is the aggregated traffic load of MR $j$ in IGW domain $i$ and $n_i$ is the number of MRs in $i$th IGW domain $D_i$.

$\beta(m)$ defined by Eq. 2 is bounded between 0 and 1 and a higher value indicates a better fairness [10]. A perfect balanced utilization (with all $Load_i$ are equal) has a fairness of 1 and a totally unfair allocation (with all traffic load given to only one IGW domain) has a fairness of $1/m$ which approach 0 in the limit as $m$ approaches infinity. Eq. 2 is a continuous function and any slight change in allocation shows up in the fairness. The fairness index also has the characteristic of monotonic property. That is, for any pair of loads with different values, if another pair with a smaller difference in their values range replaces the pair, then the balance index will increase. The load balancing algorithm is aimed at improving $\beta(m)$.

### B. Load Balancing Metric in a Heterogenous WMN

The balance index defined by Eq. 2 is designed for WMN that has identical IGW capacity. In the heterogeneous WMNs, IGWs configured with different wireless interfaces (i.e., IEEE 802.11 a/b/e/g) may have different wireless throughput capacity. For instance, in a WMN with two IGWs $I_1$ and $I_2$ with different capacities $Cap_1 = 100$ and $Cap_2 = 50$ respectively. At a given moment $I_1$ and $I_2$ have the same traffic load of $Load_1 = Load_2 = 50$. According to Eq. 2, the load balance index is perfectly equal to 1. But actually $I_1$ still has a lot of available bandwidth to handle additional traffic while the $I_2$ has reached its upper capacity limit. To take heterogeneity into account, we define the bandwidth utilization ratio $\lambda_i$ of a domain $D_i$ with $n_i$ routers, which is the percentage of utilized bandwidth as compared with the total available throughput capacity of an IGW $I_i$:

$$\lambda_i = \frac{Load_i}{Cap_i} = \frac{\sum_{j=1}^{n_i} load_{i,j}}{Cap_i}, \quad (3)$$

Based on $\lambda_i$, the fairness index in heterogeneous WMN is defined as follows:

$$\beta'(m) = \frac{\sum_{i=1}^{m} \lambda_i^2}{m \sum_{i=1}^{m} \lambda_i^2} = \frac{\sum_{i=1}^{m} Load_i/Cap_i}{\sum_{i=1}^{m} (Load_i/Cap_i)^2}. \quad (4)$$

$\beta'(m)$ will approach 1 when the bandwidth utilization ratio (i.e. $\lambda_i$) among IGW domains is closer to each other.

### A. IGW Domain Load Measure and Exchange

In the first phase, the traffic load in each IGW domain is measured and exchanged. These values are used as input to the following phases to detect load imbalance and make load migration decisions.

Each IGW estimates the traffic load value in its domain by monitoring the amount of traffic passing through it during a recent time window (e.g., $T$ seconds). Each IGW also needs to collect load information from neighboring IGW domains. Since IGWs have wired connection to Internet, they may setup high speed connection between each other to exchange the load and management information. At a given time interval, each domain can periodically broadcast the traffic load information to its neighboring domains, which enables each IGW to obtain load information of its neighboring domains. These information will be used in the following steps.

### B. Load Balancing Profitability Determination

Local domains set $Local(i)$ is defined as the set of neighboring domains of $D_i$ plus itself, i.e., $Local(i) = \{Neighbor(i) \cup i\}$. An IGW $I_1$ can calculate the local load fairness index $\beta_{local}(i)$ based on Eq. 2 when IGWs have the same configurations (or Eq. 4 for heterogeneous cases).

$$\beta_{local}(i) = \frac{\sum_{k=1}^{\|Local(i)\|} Load_k^2}{\sum_{k=1}^{\|Local(i)\|} \|Local(i)\| Load_k^2}. \quad (5)$$

Instead of greedily pursuing absolute load balancing among domains, this step determines whether to trigger the load redistribution based on a load balance threshold $\beta_{threshold}$. The load balancing process is invoked only when $\beta_{local}(i) < \beta_{threshold}$. This threshold can be a fixed value or a variable one which will be adjusted dynamically. Intuitively, the higher threshold will lead to a better load fairness, with the cost of more frequent load redistribution operations. Thus, selection of $\beta_{threshold}$ is to obtain a tradeoff between the degree of global load balancing and the communication overheads associated with migrating traffic.

### C. Load Balancing Strategy Based on Diffusion Methodology

The goal of the inter-domain migration strategy phase is to determine the source and the destination IGWs for the MRs migration. The diffusion based load balancing is an analogy between the heat diffusion phenomenon in physics and the migration of traffic load among IGW domains in which the MR migration can be interpreted as traffic diffusion among IGW domains [11], [12]. To achieve this, a portion of the MRs of the overloaded domains has to be switched to under loaded IGWs. Since in general this approach may not provide an immediate balanced solution, the process is iterated until the load balancing level among a local area reaches an acceptable value (i.e., $\beta_{local}(i) \geq \beta_{threshold}$) or some iterations limit has reached. Based on the kernel principle
used in the diffusion methods, three diffusion algorithms are proposed: basic diffusion algorithm, search unbalance domains diffusion algorithm and fast diffusion algorithm.

1) Basic Diffusion Algorithm: The basic diffusion algorithm depends on the sender initiated diffusion method [11], wherein each IGW \( I_i \) compares its local load average \( \text{Ave}_i \) with each of its neighbors’ load \( \text{Load}_{j} \), and then switches some MRs to the neighboring domains with load below the average (called deficient neighbors). The following notations are used in this method. Local load average \( \text{Ave}_i \) is the average load in the local domains set \( \text{Local}(i) \):

\[
\text{Ave}_i = \frac{1}{|\text{Local}(i)|} \sum_{k=1}^{\text{Local}(i)} \text{Load}_k. \quad (6)
\]

The difference between \( \text{Load}_i \) and \( \text{Ave}_i \), represents the excess load of IGW \( I_i \), and is defined as:

\[
\text{Ex}_i = \text{Load}_i - \text{Ave}_i. \quad (7)
\]

Each neighboring \( D_j \) is assigned a deficiency value \( D_{j,i} \), based on following equations:

\[
D_{j,i} = \begin{cases} 
\text{Ave}_j - \text{Load}_j, & \text{if } \text{Load}_j < \text{Ave}_i, \\
0, & \text{otherwise}.
\end{cases} \quad (8)
\]

These values are further added to obtain the total deficiency,

\[
\text{TD}_i = \sum_{j=1}^{\text{Neighbor}(i)} D_{j,i}. \quad (9)
\]

Thus, we can get the portion of IGW \( I_i \)'s excess load which is apportioned to neighbor \( D_j \),

\[
P_{i,j} = \frac{\text{Ex}_i \times D_{j,i}}{\text{TD}_i} = (\text{Load}_i - \text{Ave}_i) \times \frac{D_{j,i}}{|\text{Neighbor}(i)|} \sum_{j=1}^{\text{Neighbor}(i)} D_{j,i}. \quad (10)
\]

Based on the above definitions, the basic diffusion algorithm is defined as in Algorithm 1.

Function \( \text{GetNeighbor}(G) \) gets the neighboring IGW domains of \( D_i \), and \( \text{CalLocalFairIndex}(D_i, \text{Neighbor}(i)) \) calculates the local load fairness index based on Eq. 5. \( \{P_{i,j}\} \) is the integer part of \( P_{i,j} \), which represents the traffic load units to be migrated from \( D_i \) as MR migration are always in discrete units.

**Algorithm 1: Basic Diffusion Algorithm**

```
Input: G = {V, E}, Load = {Load_1, ..., Load_m}, D = {D_1, ..., D_m}, \beta_{threshold}

Output: Load, D

\text{Neighbor}(i) = \text{GetNeighbor}(G, i)
\beta_{local} = \text{CalLocalFairIndex}(D_i, \text{Neighbor}(i))

if \beta < \beta_{threshold} then
  for j \in \text{Neighbor}(i) do
    Load_i = Load_i - \lfloor P_{i,j} \rfloor
    Load_j = Load_j + \lfloor P_{i,j} \rfloor
    Migrate \{P_{i,j}\} units traffic from \( D_i \) to \( D_j \)
  end
end
```

2) Search Unbalanced Domain Diffusion Algorithm: However, the basic diffusion algorithm discussed above has two disadvantages. First, the number of iterations required by the load balancing algorithm may be too high. As showed in [12], a non-full connected square topology with four nodes need 10 iterations steps to reach stable state. The second problem is that the algorithm may produce solutions which is locally balanced although they are globally unbalanced. Such a situation is illustrated by Fig. 2, in which the IGW domain \( D_i \) is represented by a gray circle with the IGW \( I_i \) in the center. The traffic load of three neighboring domains (e.g., Fig. 2(a)) is obtained as balanced solution after applying the basic diffusion algorithm. While as showed in Fig. 2(b), there exists a better load balance with a higher fairness index 1. Then, an improved diffusion algorithm, called search unbalanced domain (SUD) diffusion algorithm is defined. The idea is to do dynamic load balancing strategy not only comparing the traffic load between the local load average with neighbors, also try to minimize the maximum load difference between any two neighboring domains [12].

In addition to the notations defined in the basic diffusion algorithm, below notations will be used in the algorithm. First, the maximum difference between two IGW domains in the IGW neighborhood area is \( \text{MaxDiff}_{i,j} \), i.e.,

\[
\text{MaxDiff}_{i,j} = \max(L_k - L_j) \quad \forall k, j \in \text{Local}(i). \quad (11)
\]

Neighboring traffic load average \( \text{Ave}'_i \) is load average of the neighboring IGW domains of \( D_i \):

\[
\text{Ave}'_i = \frac{1}{|\text{Neighbor}(i)|} \sum_{k=1}^{\text{Neighbor}(i)} \text{Load}_k. \quad (12)
\]

and neighbor load deviation \( \sigma_i \) is defined as

\[
\sigma_i = \sqrt{\sum_{k=1}^{\text{Neighbor}(i)} |\text{Ave}'_i - \text{Load}_k|^2}. \quad (13)
\]

Based on these definitions, the SUD diffusion algorithm are described as in Algorithm 2.

Algorithm 2 solves the problems as in Fig. 2 (a) and output the optimal balancing Fig 2 (b). While it still required many iterations before the final status is arrived in large networks, which will lead to the flapping of MR migration due to the aging of information [11].
Algorithm 2: SUD Diffusion Algorithm

Input: $G = (V,E)$, $Load = \{Load_1, \cdots, Load_m\}$, $D = \{D_1, \cdots, D_m\}$, $\beta_{threshold}$

Output: $Load$, $D$

$Neighbor(i) = \text{GetNeighbor}(G, i)$

$\beta_{local} = \text{CalLocalFairIndex}(D_i, Neighbor(i))$

if $\beta < \beta_{threshold}$ then
  if $Ex_i > 0$ then
    for $i_j \in Neighbor(i)$ do
      $Load_i = Load_i - \lfloor P_{i,j} \rfloor$
      $Load_j = Load_j + \lfloor P_{i,j} \rfloor$
      Migrate $\lfloor P_{i,j} \rfloor$ units traffic from $D_i$ to $D_j$
    end
  else
    if $MaxDiff_i > 1$ then
      if $\sigma_i = 0$ then
        Distribute $[Load_i - (Ave_i + 1)]$ among $Neighbor(i)$
      else
        Migrate 1 unit traffic to lowest load neighbor
      end
    end
  end
end

3) Fast Diffusion Algorithm:

Algorithm 3: Fast Diffusion Algorithm

Input: $G = (V,E)$, $Load = \{Load_1, \cdots, Load_m\}$, $D = \{D_1, \cdots, D_m\}$, $\beta_{threshold}$

Output: $Load$, $D$

$Neighbor(i) = \text{GetNeighbor}(G, i)$

$\beta_{local} = \text{CalLocalFairIndex}(D_i, Neighbor(i))$

if $\beta < \beta_{threshold}$ then
  if $Ex_i > 0$ then
    for $i_j \in Neighbor(i)$ do
      $LocalDiff_{i,j} = Load_j - Ave_i$
      if $LocalDiff_{i,j} > 0$ then
        $Load_i = Load_i + \lfloor LocalDiff_{i,j} \rfloor$
        $Load_j = Load_j - \lfloor LocalDiff_{i,j} \rfloor$
        Migrate $\lfloor LocalDiff_{i,j} \rfloor$ units traffic from $D_i$ to $D_j$
      else
        $Load_i = Load_i - \lfloor LocalDiff_{i,j} \rfloor$
        $Load_j = Load_j + \lfloor LocalDiff_{i,j} \rfloor$
        Migrate $\lfloor LocalDiff_{i,j} \rfloor$ units traffic from $D_i$ to $D_j$
      end
    end
  else
    if $MaxDiff_i > 1$ then
      if $\sigma_i = 0$ then
        Distribute $[Load_i - (Ave_i' + 1)]$ among $Neighbor(s(i))$
      else
        Migrate 1 unit traffic to lowest load neighbor
      end
    end
end

The fast diffusion load balancing algorithm method is illustrated in Algorithm 3, which consists of two phases. In the first phase, the local load average of domain $D_i$ (i.e., $Ave_i$) is calculated based on equation 6. Then, each neighbor’s load is compared with $Ave_i$ to get the local difference $LocalDiff_{i,j}$, i.e., $LocalDiff_{i,j} = Load_j - Ave_i$. The excess traffic in the overloaded domains is migrated to the under loaded domains based on $LocalDiff_{i,j}$. In the second phase of the algorithm, the similar strategy as in SUD algorithm is used to minimize the maximum difference among neighboring domains.

Based on the fast diffusion algorithm, we can not only overcome the shortcoming of the basic diffusion algorithm illustrated in Fig. 2, but also achieve load balancing at a faster rate as compared with the previous two algorithms. These is verified by simulations in the next section.

D. Inter-domain MR Migration

The load balancing algorithm outputs updated traffic load distribution. Then, the IGW domains switch some MRs between neighbor domains based on the initial and final load status. The process should also guarantee that the migration will not violate the MRs and IGW capability constraints. Since the domain load may change irregularly and randomly, it is difficult to maintain an absolute balance between two adjacent domains. Migrations stops when there are no available MRs to migrate or the local load balancing can not be improved any further.

VI. PERFORMANCE EVALUATION

To understand the impact of proposed load balancing scheme, we performed simulation-based analysis on the three diffusion based load balancing approaches. For ease of comparison, the simulation is based on a 5 by 5 grid topology of 25 IGW domains, which regulates the maximum neighbors of any IGW domain to be four. Note that the grid topology is not a necessary condition for our algorithms and can easily apply to any random network. We assume densely deployed mesh nodes that each IGW domain has enough bordering MRs to be switched to four adjacent domains. We assume the IGWs to have uniform throughput capacity of 50 units and the traffic in each IGW domains are randomly generated between 0 and 50 units. The global load balancing index is evaluated based on Eq. 2. The original load balance level is evaluated as $\beta_{original}$. The new load balance level $\beta_{new}$ and the improvement from the load balancing procedure (i.e., $\Delta \beta = \frac{\beta_{new} - \beta_{original}}{\beta_{original}} \times 100\%$), are computed. In each evaluation, the simulation is run 50 times with randomly generated traffic. The average results are shown as the final results.

A. Effects of Load Balance Threshold

First we evaluate the impact of selecting load balancing threshold $\beta_{threshold}$ on global load fairness level. The load balance threshold $\beta_{threshold}$ is increased from 0.7 to 1 and the number of iterations is limited to 10. As shown in Fig. 3, when the load balance threshold is increased, the final level is improved after the load balancing procedure. Fig. 3 also shows that the fast diffusion algorithm provides better load distribution than the other two algorithms. When the $\beta_{threshold} = 0.9$, the final load balance level can be improved by 30%, 25% and 20% by the fast diffusion algorithm, SUD diffusion algorithm and basic diffusion algorithm respectively.
B. Effects of Load Balancing Iterations

Since the load balancing strategies are distributed, and each IGW initiates the load redistribution based on the trigger principle, i.e., $\beta_{local} < \beta_{threshold}$. In each iteration, every IGW checks for local load balance level and compares it with the threshold $\beta_{threshold}$, which is set to 1 here. The iterations number is varied from 1 to 10. Fig 4 shows that larger number of iterations increases the resulting load balance level. Among these three load balancing algorithms, the fast diffusion algorithm only needs two to three iterations to get the near optimal load fairness level 1, and with an improvement $\Delta \beta$ around 35%. The SUD diffusion algorithm can arrive at the same fairness level by taking 8 or 9 iteration steps. Due to limitations of the basic diffusion algorithm, it reaches stable state after six iterations with a lower fairness level compared with the other two algorithms.

VII. CONCLUSIONS

In this paper, the load balancing among different IGW domains in a WMN is discussed. The load balancing indices are defined for both homogenous and heterogeneous WMNs. To have a better global load balancing, a distributed load balancing scheme is proposed. Load balancing algorithms based on three diffusion methods are introduced in the proposed scheme. By dynamically migrating MRs between neighboring IGW domains, the traffic load distribution in different domains can arrive at a relatively better fairness after few iterations. The simulation results confirms proposed load balancing scheme to have good scalability, reliable robustness and fast convergence.

ACKNOWLEDGMENT

This paper is supported by STTR project MeshXpress II under AFOSR support.

REFERENCES


