On Optimal Distributed Channel Allocation for Access Points in WLANs

Tânia L. Monteiro[1], Marcelo E. Pellenz[1], Manoel C. Penna[1], Fabrício Enembreck[1], and Richard Demo Souza[2]

[1] PPGIa, PUCPR, Curitiba - PR, Brazil
{taniamonteiro,marcelo,penna,fabricio}@ppgia.pucpr.br
[2] CPGEI, UTFPR, Curitiba - PR, Brazil
richard@utfpr.edu.br

Abstract. We propose a new distributed algorithm for optimal channel assignment in WLANs with multiple access points, applying a novel formulation for wireless networks based on Distributed Constraint Optimization Problem (DCOP). The DCOP approach allows to model a wide variety of distributed reasoning tasks of multi-agent applications. The proposed strategy is derived from a polynomial-space algorithm for DCOP named ADOPT, which is guaranteed to find the global optimal solution while allowing agents to execute asynchronously and in parallel. Our proposed algorithm, denoted DCAA-O, allows a group of APs to coordinate themselves in order to find the optimal channel allocation solution which minimizes the network interference. The algorithm performance is evaluated in terms of the required number of transmitted control messages among APs. It is shown that DCAA-O outperforms a recently proposed channel assignment strategy for WLANs.

Keywords: Wireless Networks, Channel Assignement, Distributed Constraint Optimization Problem.

1 Introduction

Typical deployments of Wireless Local Area Networks (WLANs) utilize multiple access points (APs) equipped with a single radio interface. Such networks can suffer from serious capacity degradation due to the half duplex nature of the wireless medium and also due to the interference between simultaneous neighboring transmissions. Fortunately the IEEE 802.11b/g and 802.11a standards [1], provide 3 and 12 non-overlapping channels, respectively, which could be used simultaneously within a neighborhood [2]. The ability to utilize multiple channels, benefiting from the whole available spectrum, substantially increases the capacity of wireless networks [3]. The implementation of a dynamic and distributed algorithm for channel allocation is fundamental to keep the network operating at its maximum capacity. Several methods have been proposed for dynamic spectrum access, including pricing and auction mechanisms and graph coloring. Some promising cooperative strategies for channel assignment have been recently proposed in the literature for application in WLANs [4] and mainly in wireless mesh networks (WMNs) [5, 6]. Specifically, our study about channel assignment focus on

WLAN scenarios. The existing distributed algorithms do not guarantee optimal solution and may present convergence instability. As many WLAN scenarios are typically not large enough to make an optimization approach unfeasible, we identify the lack of an optimal distributed strategy for channel allocation in order to reduce interference between APs in the same physical neighborhood. Additionally, we consider that APs in the same transmission range can opportunistically communicate among themselves over wireless channel to coordinate a distributed optimization process that guarantee the best solution for the experienced interference scenario. Such coordination approach over wireless channel allows the proposed algorithm to be applied to a wireless network formed by APs belonging to different WLANs.

The paper presents two main contributions. The first one is the novel formulation for the channel assignment problem as a DCOP and the second one is the proposal of a new distributed and synchronous algorithm called DCAA-O (Distributed Channel Assignment Algorithm - Optimal), which is able to reach the optimal frequency assignment, irrespective of the network topology, number of APs, network density and available channels. We compare DCAA-O with a recently proposed measurement-based algorithm, *Local-Coord* (LO-A), presented by Chen et al. in [4]. It was shown in [4] that LO-A performs better than algorithms proposed in [7–10]. The remainder of this paper is organized as follows. Section 2 describes DCOP and ADOPT algorithm [11]. In Section 3 we present the novel formulation for channel assignment in WLANs via DCOP. The proposed algorithm is described in Section 4. The evaluation scenario and results are presented in Sections 5 and 6. Section 7 concludes the paper.

2 Background

In artificial intelligence research area, a *distributed constraint optimization problem* (DCOP), consists of a set of variables that are distributed to a group of collaborative agents as *valued* constraints, that is, constraints that are described as valuable functions that return values in a specific range. The goal is to optimize a global objective function, maximizing the weight of satisfied constraints [11].

Consider a set of N agents, $A = \{a_1, a_2, \ldots, a_N\}$, and a set of N values, $D = \{d_1, d_2, \ldots, d_N\}$, where each value d_j is assigned to an agent a_j and belongs to a finite discrete domain $\mathbb{D}_j$. For each pair of agents (a_i, a_j) a cost function $f_{ij}(x, y) : \mathbb{D}_i \times \mathbb{D}_j \longrightarrow \mathbb{R}$ is defined. Agents have to coordinate themselves in order to find a set of values that optimize a global function establishing the costs for constraints. Therefore, the goal of a DCOP algorithm is to find the optimal set, denoted D^*, whose values minimize a global cost function $g^* = g(D^*)$.

Modi et al. [11] proposed Simple-ADOPT, the first algorithm for DCOP that can find the *optimal* solution, using only localized asynchronous communication and polynomial space at each agent. Simple-ADOPT is a backtracking search algorithm that is executed asynchronously and in parallel on every agent, updating lower bounds on global solution quality. The Simple-ADOPT algorithm requires agents to be prioritized in a total order, making use of a *spanning tree* (ST), obtained through Depth-First Search (DFS)[12]. The authors use the term *parent* to refer to an agent's immediate higher priority agent in the ordering and *child* to refer to an agent's immediate lower priority agent in the ordering. Figure 1b shows the total order on agents for constraint graph of Figure 1a, represented by the ST generated by the DFS algorithm. Giving this ordering, an agent communicates its value d_j to all linked lower priority agents, as indicated in Figure 1c, and communicates a lower bound cost to a unique higher priority agent as indicated in Figure 1d.

In order to change its value and lower bound cost information between agents, the Simple-ADOPT algorithm employs two types of control messages, denoted VALUE and VIEW messages respectively. The flow direction of VALUE and VIEW messages are also indicated in Figures 1c and 1d, respectively. These graphs are required input parameters for the Simple-ADOPT algorithm and they will be denoted as *VALUE graph* and *VIEW graph*, for short. Let P_j^{value} and C_j^{value} be the set of parents and children of an agent a_j in the *VALUE graph*, respectively. Likewise, let P_j^{view} and C_j^{view} be the set of parents and children of an agent a_j in the *VIEW graph*. A pair agent/value of the form (a_j, d_j) is called a *view*. The variable $Currentvw_j$ of an agent a_j is the current set of *views* $\{(a_{i1}, d_{i1}), (a_{i2}, d_{i2}), \ldots, (a_{il}, d_{il})\}$, composed by the all linked ancestor's agent/value pairs, which means that $\{a_{i1}, a_{i2}, \ldots, a_{il}\} \in P_j^{value}$. The variable vw_j of an agent a_j stores the current view of agent a_j regarding his parent agent $a_i \in P_j^{view}$ in the VIEW graph, denoted $(a_i, d_i)_j$. Therefore $vw_j = (a_i, d_i)_j \in Currentvw_j$ of agent a_j. The variable $Currentvw_j$ is sent to his parent agent $a_i \in P_j^{view}$ using a VIEW message. When a parent agent a_i receives the VIEW message containing $Currentvw_j$ from child agent a_j, it must perform a view compatibility test. We say that two views ($Currentvw_j$ and $Currentvw_i$) are compatible if the views vw associated to the same parent nodes in the VALUE graph have the same value d, which means that $\forall vw = (a, d) \in Currentvw_j$ we also have $vw \in Currentvw_i$.

Simple-ADOPT begins by each agent a_j choosing locally and concurrently a value $d_j \in \mathbb{D}_j$. This value is sent to all its linked descendent agents $\{a_{k1}, a_{k2}, \ldots, a_{kn}\} \in C_j^{value}$, using VALUE messages. Then agents asynchronously wait for and respond to incoming messages. Given the received VALUE messages from parents agents, agent a_j chooses his local value d_j according to (1).

$$d_j = x \mid \min_{x \in \mathbb{D}_j} \sum_{a_i \in P_j^{value}} f_{ij}(d_i, x) \; . \tag{1}$$

During the Simple-ADOPT algorithm execution, each agent a_j must deal with three distinct cost values, the *local cost*, the *current lower bound cost* and the *estimated lower bound cost*. The *local cost* is defined by $lc_j = \sum_{a_i \in P_j^{value}} f_{ij}(d_i, d_j)$. It is computed as sum of cost functions for all VALUE messages received from ancestors, using the chosen value d_j obtained from (1). The *current lower bound cost* for subtree is computed according to $sc_j = \sum_{a_k \in C_j^{view}} z_k^*$. It is the sum of all *estimated lower bound costs* received from children agents in the VIEW graph, $\{a_{k1}, a_{k2}, \ldots, a_{kn}\} \in C_j^{view}$. The last one is the *estimated lower bound cost* defined by $z_j^* = lc_j + sc_j$, which is the sum of local cost lc_j with the cost sc_j. If an agent a_j does not have any descendent, his *estimated lower bound cost*, z_j^*, is just its local cost lc_j. Whenever an agent a_j receives a VALUE message from a *linked ancestor* $a_i \in P_j^{value}$, it stores the current received value (a_i, d_i) into his $Currentvw_j$ variable, which represents a_j's current context. For

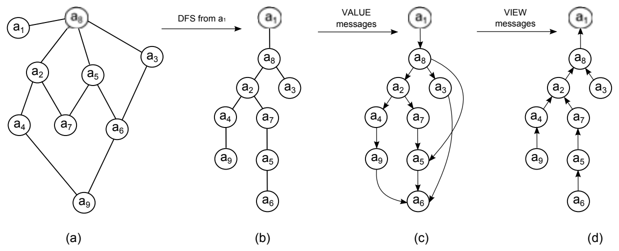

Fig. 1. DCOP Example: (a) Constraint graph; (b) Constraint graph of totally ordered agents; (c) Graph of VALUE messages; (d) Graph of VIEW messages

its current context, the agent a_j then reports the estimated lower bound cost z_j^* to his parent agent $a_i \in P_j^{view}$, by a VIEW message. In its turn, his parent agent a_i, will use the received *estimated lower bound cost* z_j^* in order to computed its *current lower bound cost for subtree*, sc_i. The flow of VIEW messages is different from the flow of VALUE messages. The VALUE graph has its origin on the DFS tree plus all conflict edges existing between vertices in the undirected constraint graph $\mathbf{G} = (V, E)$ that is not represented in the DFS tree. Meanwhile the VIEW graph is exactly the DFS tree. The parent agent receives a VIEW message but throw away this message when $vw_j \notin Currentvw_i$. This situation can happen in two cases: either the parent agent a_i has a more up-to-date current view than its child a_j or a_j has a more current up-to-date view than its parent a_i. When an agent a_j reports to its parent P_j^{view}, its estimated lower bound cost z_j^*, locally the agent assumes the value d_j that has minimized z_j^*. Finally, when an agent a_j receives a VALUE message, it updates its current context by updating the current values d_i of its parents. The agent then deletes its stored cost z_j^*, since it may now be invalid. Simple-ADOPT by storing only one current view, has linear space requirements at each agent, considering the number of variables that an agent can assume, represented by the finite discrete domain $\mathbb{D}_j$. The algorithm reaches a stable state when all agents are waiting for incoming messages. Then the complete assignment chosen by the agents is equal to the optimal assignment set, denoted D^*, whose values minimize a global cost function, $g^* = g(D^*)$.

3 Modeling Channel Assignment via DCOP

In WLANs a large number of access points may be operating in the same region, leading to a scenario where the *management* of wireless channels is extremely important because *co-channel* and *adjacent-channel* interference can significantly reduce the network performance. Consider the problem of optimizing the throughput of a WLAN with multiple APs. This requires that channels of all APs have to be set to maximize the signal-to-interference ratio. As the number of possible channels grows exponentially with the number of APs, we have a NP complex combinatorial optimization problem [13].

The approach of this work is to propose a distributed synchronous parallel method based on multi-agents concept [15], allowing any set of APs coordinate for searching

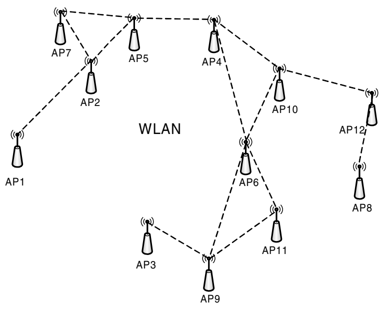

Fig. 2. Example of a physical WLAN deployment

the optimal solution for the channel allocation problem. Since the domains are inherently distributed, we require a method where agents can optimize a global function in a distributed way, using only local coordination (communication with neighboring agents). As our challenge is not only to ensure computational time but also to ensure an acceptable amount of exchanged messages between agents, we adopted a distributed synchronous and parallel solution, that unlike some conventional distributed approaches as Asynchronous Backtracking (ABT) [14] and the Synchronous Branch and Bound (SynchBB) algorithm in [16], ensures the optimal solution. The considered environment refers to one or more stationary wireless networks, composed by APs arbitrarily distributed over an area, where each AP is equipped with a single IEEE802.11b radio interface [1]. We consider the reduced problem where all radio interfaces across the network use omnidirectional antennas and have identical transmission powers. The goal is find the best channel allocation solution for each AP in order to improve the overall network performance.

In order to apply DCOP an undirected *constraint graph* $\mathbf{G} = (V, E)$ should be obtained from the physical network topology. Consider as an example a WLAN scenario composed of 12 APs, whose physical layout is shown in Figure 2. The dotted lines in Figure 2 represent the connectivity between APs. This means that their respective channel allocations will affect the performance of both due to the multiple access contention and interference. Therefore, the graph represented by the connections between APs in Figure 2 constitutes the *constraint graph* for the channel allocation problem. In such case, the set of vertices V of the constraint graph $\mathbf{G}$ represents the APs, and the set of conflict edges $E \subseteq V^2$ is represented by connectivity links between APs. As described in Section 2 agents must be prioritized in a total order, making use of a *spanning tree*. An initialization procedure using a predefined common control channel is required allowing each AP to identify the APs in the neighborhood (constraint graph) and also to configure his logical ordering in the VALUE and VIEW graphs. This can be accomplished by means of a distributed spanning tree algorithm. After the logical graph construction each AP knows its parent, child, and linked descendants. Following the approach used in [13] we consider a cost function metric based on channel spectrum overlapping. Specifically in North America and Europe, there are 11 different channels that an AP may select in IEEE802.11b/g 2.4GHz ISM band. This ISM band is allocated between 2400 and 2483MHz, with channel central frequencies starting at

2412MHz and spaced at 5MHz intervals to 2462MHz. As the channel bandwidth is about 22MHz, there is a substantial spectrum overlap between adjacent channels. If adjacent APs select overlapping channels, it will be induced a spectral interference between concurrent transmissions or multiple access contention. An accurate interference model is necessary taking into account the level of spectrum overlap. For instance, a 5.5 or 11Mbps DSSS (Direct Sequence Spread Spectrum) unfiltered modulated signal has a power spectrum behavior described by:

$$\mathcal{S}(n,f) = \begin{cases} \left| \frac{\sin[2\pi\mathcal{X}(n,f)]}{2\pi\mathcal{X}(n,f)} \right|^2, & \text{for } \mathcal{X}(n,f) \neq 0 \\ 1, & \text{for } \mathcal{X}(n,f) = 0 \ . \end{cases} \tag{2}$$

where $\mathcal{X}(n,f) = \frac{f-2412-5(n-1)}{BW}$, BW is the null-to-null channel bandwidth and n is the channel number ($n = 1, 2, \ldots, 11$). The WLAN cards employ an IF filtering on both transmit and receive paths in order to reduce sidebands power. The approximated IF filter power spectrum response is defined by $\mathcal{F}(n,f) = [1 + 2.6\ \mathcal{X}(n,f)]^{-6}$ [13]. It has 3dB bandwidth of 17MHz and stopband 50dB down at ±22MHz. From (2) we can now determine the channel overlap factor, which determines the co-channel interference level between two APs. The spectrum overlap factor $\mathcal{SO}(n,m)$ between the channel number n and the channel number m is given by (3). The overlap factor can be normalized by the maximum overlap factor which occurs when both channels are the same. The normalized factor is defined by (4) as a function of the channel spacing, $c_{spc} = n - m$, where $n \geq m$ and $c_{spc} \in \{0, 1, 2, \ldots, 10\}$. Table 1 presents the normalized overlap factor $\mathcal{SO}_{norm}$ as a function of channel spacing c_{spc}, obtained using (4). Its is important to point out that the numerical results shown in Table 1 may vary for different chipset and filter configurations of the WLAN card.

$$\mathcal{SO}(n,m) = \int_{2200}^{2700} \mathcal{S}(n,f)\mathcal{F}(n,f)\mathcal{S}(m,f)\mathcal{F}(m,f)\ df \ . \tag{3}$$

$$\begin{aligned} \mathcal{SO}_{norm}(c_{spc}) = \frac{1}{\mathcal{SO}(1,1)} \int_{2200}^{2700} & \mathcal{S}(1,f)\mathcal{F}(1,f) \\ & \mathcal{S}(c_{spc}+1,f)\mathcal{F}(c_{spc}+1,f)\ df \ . \end{aligned} \tag{4}$$

In our DCOP formulation, the selected value d_j of an agent a_j represents the channel selection number of the AP$_j$, therefore the domain of d_j is defined as $\mathbb{D}_j = \{1, 2, \ldots, 11\}$. For a pair of connected agents (pair of interfering APs) in the constraint graph, the cost function for a pair of values (d_i, d_j) is defined as $f_{ij}(x,y) : \mathbb{D}_i \times \mathbb{D}_j \to \mathbb{R}$ where the values in $\mathbb{R}$ represent the interference overlap factors of Table 1. The APs by coordinating themselves have to find a set of channel allocation values, D^*, that optimizes a global function establishing the costs for constraints that minimize the total amount of interference. In practice, the channel overlap factor can be weighted by a normalized path loss weighting factor denoted by α_{ij} where $0 \leq \alpha_{ij} \leq 1$, resulting in a local cost, $\alpha_{ij}\ f_{ij}(x,y)$, between the APs. The parameter α_{ij} can be computed by agent a_i based on the received signal strength intensity (RSSI) of the received message from agent a_j. In this study we consider the worst case interference scenario where $\alpha_{ij} = 1$.

Table 1. Adjacent Channel Interference Factor

Channel Spacing	Overlap Factor
0	1
1	0.7272
2	0.2714
3	0.0375
4	0.0054
5	0.0008
6	0.0002
7 - 10	0

4 Proposed Algorithm

The Simple-ADOPT algorithm operates asynchronously and in parallel on every agent. In many multi-agent applications this is not a restriction but, in our case, this is an undesirable characteristic because we want to minimize the number of overhead messages among APs. Therefore, we derived DCAA-O from Simple-ADOPT keeping the optimality conditions and imposing synchronous message control at agents. Our algorithm employs the same basic notation for parents and children nodes, as well as the definition of VALUE and VIEW graph. The pseudocode of the proposed DCAA-O method is shown in Algorithm 1. The parameters listed below are also based on the description of ADOPT, however some changes were necessary for the development of our algorithm.

a_j: Current Access Point;

a_1: Root Access Point;

A'_i: Set of APs descendent of an ancestor $a_i \in P_j^{value}$;

$P_j^{value'} = P_j^{value} \cap A'_i$: Subset of P_j^{value} whose nodes are descendent of an ancestor $a_i \in P_j^{value}$;

Currentvw$_j = \{(a_i, d_i^t)\}, \forall a_i \in P_j^{value}$: Current view of AP a_j;

LocalCosts$_j = \{[d_j^t, lc_j(d_j^t), sc_j(d_j^t)]\}, \forall d_j^t \in \mathbb{D}_j$: Temporary Local Costs of AP a_j for each choice d_j^t

SubTreeValues$_j(d_j^t) = \{(a_j, d_j^t), (a_k, d_k), \ldots\}, \forall a_k \in$ subtree of a_j : Temporary subtree costs;

$lc_j(d_j^t)$: Local cost of AP a_j given d_j^t and **Currentvw**$_j$;

$sc_j(d_j^t) = \sum_{a_k \in C_j^{view}} z_k(d_j^t)$: Subtree cost of a_j;

$z_j(d_j) = lc_j(d_j) + sc_j(d_j)$ where $d_j = x \mid \min_{x \in \mathbb{D}_j} lc_j(x) + sc_j(x)$;

The choice of the value d_j is an important point of the distributed algorithm execution. A given node a_j will iterate with its subtree, through its VALUE and VIEW children, for different temporary $d_j^t \in \mathbb{D}_j$ choices. During the iteration, the node a_j builds its variables, **LocalCosts**$_j$ and **SubTreeValues**$_j$. After testing all d_j^t, the node is able to identify the value d_j that minimizes the cost of the subtree in the context represented by $z_j(d_j)$. From this time the node a_j sends a VIEW message to its parent $a_i \in P_j^{view}$, that can continue with its procedure for choosing its local channel d_i. The DCAA-O algorithm can initiate the channel reallocation in a network composed of

Algorithm 1. DCAA-O

```
1  Procedure Initialization
2      $\mathbf{Currentvw}_j = \emptyset$;
3      $\mathbf{LocalCosts}_j = \emptyset$;
4      $\mathbf{SubTreeValues}_j(d_j) = \emptyset \ \forall d_j \in \mathbb{D}_j$;
5  end
6  Procedure RootNode
7      forall $(d_1^t \in \mathbb{D}_1)$ do
8          Send_VALUE_Message($a_1$, $d_1^t$) $\forall a_k \in C_1^{value}$;
9          forall $(a_k \in C_1^{view})$ do
10             Receive_VIEW_Message $(a_k, z_k(d_k), \mathbf{SubTreeValues}_k(d_k))$;
11             $\mathbf{SubTreeValues}_1(d_1^t) = \mathbf{SubTreeValues}_1(d_1^t) \cup \mathbf{SubTreeValues}_k(d_k)$;
12             $sc_1(d_1^t) = sc_1(d_1^t) + z_k(d_k)$;
13         end
14         Add $[d_1^t, 0, sc_1(d_1^t)]$ to $\mathbf{LocalCosts}_1$;
15     end
16     $d_1 = x \mid \min_{x \in \mathbb{D}_1} sc_1(x)$, where $sc_1(x) \in \mathbf{LocalCosts}_1$;
17     Send_TERMINATE($\mathbf{SubTreeValues}_1(d_1)$) $\forall a_k \in C_1^{view}$;
18 end
19 Procedure OtherNodes
20     while TERMINATE_QUEUE= $\emptyset$ do
21         Receive_VALUE_Message($a_i$, $d_i^t$);
22         Add $(a_i, d_i^t)$ to $\mathbf{Currentvw}_j$;
23         forall $(a_i \in P_j^{value'})$ do
24             Receive_VALUE_Message($a_i$, $d_i^t$);
25             Add $(a_i, d_i^t)$ to $\mathbf{Currentvw}_j$;
26         end
27         forall $(d_j^t \in \mathbb{D}_j)$ do
28             $lc_j(d_j^t) = \sum_{a_i \in \mathbf{Currentvw}_j} f_{ij}(d_i^t, d_j^t)$;
29             $sc_j(d_j^t) = 0$;
30             Add $[d_j^t, lc_j(d_j^t), sc_j(d_j^t)]$ to $\mathbf{LocalCosts}_j$;
31         end
32         if $(C_j^{value} \neq \emptyset)$ then
33             forall $(d_j^t \in \mathbb{D}_j)$ do
34                 Send_VALUE_Message($a_j$, $d_j^t$) $\forall\ a_k \in C_j^{value}$;
35                 forall $(a_k \in C_j^{view})$ do
36                     Receive_VIEW_Message $(a_k, z_k(d_k), \mathbf{SubTreeValues}_k(d_k))$;
37                     $\mathbf{SubTreeValues}_j(d_j^t) =$
                       $\mathbf{SubTreeValues}_j(d_j^t) \cup \mathbf{SubTreeValues}_k(d_k)$;
38                     Add $z_k(d_k)$ to $sc_j(d_j^t)$ in $\mathbf{LocalCosts}_j$;
39                 end
40             end
41         end
42         $d_j = x \mid \min_{x \in \mathbb{D}_j} lc_j(x) + sc_j(x)$, where $lc_j, sc_j \in \mathbf{LocalCosts}_j$;
43         $z_j(d_j) = lc_j(d_j) + sc_j(d_j)$;
44         $\mathbf{SubTreeValues}_j(d_j) = \mathbf{SubTreeValues}_j(d_j) \cup \{(a_j, d_j)\}$;
45         Send_VIEW_Message($a_j, z_j(d_j), \mathbf{SubTreeValues}_j(d_j)$) to $a_i \in P_j^{view}$;
46     end
47     Receive_TERMINATE($\mathbf{SubTreeValues}_1(d_1)$);
48     Set channel $d_j = d_m \mid a_j = a_m, (a_m, d_m) \in \mathbf{SubTreeValues}_1(d_1)$;
49     Send_TERMINATE($\mathbf{SubTreeValues}_1(d_1)$) $\forall a_k \in C_j^{view}$;
50 end
```

multiple APs, when an AP identifies co-channel or adjacent channel interference. In a preliminary step the AP a_j initiates a distributed algorithm for calculating the spanning tree [17, 18] that allows the definition of VALUE and VIEW graphs on all nodes. After the execution of the algorithm, as presented in Algorithm 1, the root node sends a message identified as *TERMINATE*, containing optimal allocation solution for all other APs through the graph VALUE.

The algorithm DCAA-O was implemented and simulated in a distributed way, using a library of discrete event simulation denoted *SIMPATICA*[19], based on the actors/messages paradigm. According to this paradigm, a simulation model is composed by a set of actors (or tasks) that communicate among them using messages. This library allowed us to simulate a distributed synchronous environment through three kinds of entities: task, queue and message. Each AP is implemented as a task and has three queues for receiving messages, VIEW_QUEUE, VALUE_QUEUE, and TERMINATE_QUEUE, respectively.

5 Evaluation Scenario

The performance of DCAA-O was evaluated on random generated network topologies. We considered topologies of 4, 9 and 16 nodes, with graph densities of 1, 2, 3, 4 and 5, when applicable. A graph with *link density* d has $d \cdot n$ links, where n is the number of nodes in the graph [11]. For a given number of nodes, a total of 30 random topologies was generated for each density. As a first study scenario, we allowed the APs to select the corresponding channel from an available set with only three channels, composed of the tree *non-overlapping* channels $\{1, 6, 11\}$. In the second scenario APs can select one among 11 available channels. We compared the proposed algorithm to the scalable distributed algorithm, denoted *Local-Coord* (LO-A), recently proposed by Chen at al. [4]. LO-A is a simple distributed algorithm, featuring a good balance between solution quality and number of messages exchanged between the APs. However to their considered environment, an AP needs to locally coordinate with others APs via a wired backbone network for channel switching. For comparison purposes, LO-A messages are exchanged among APs via a pre-defined wireless control channel as for DCAA-O.

6 Simulation Results

The performance of DCAA-O was evaluated in terms of the average number of exchanged messages between APs for convergence to optimal solution. The LO-A algorithm was used as a benchmark algorithm. The stopping criterion for LO-A algorithm concerns to a minimum number of iterations without any improvement or new channel allocation among the APs. We considered a lower bound of 50 iterations for LO-A. This value was determined by simulation as an appropriate criterion to stop the execution of the LO-A algorithm. Table 2 presents the comparison results for topology scenarios with 4 APs. In this case the random generated network topologies may only have graph density one. The DCAA-O algorithm presented a number of exchanged messages at least 90% smaller than LO-A. The final global cost value obtained by both algorithms

Table 2. Average Number of Messages - Topologies with 4 APs

Available Number of Channels	DCAA-O	LO-A
3 Channels	29	1324
11 Channels	277	2140

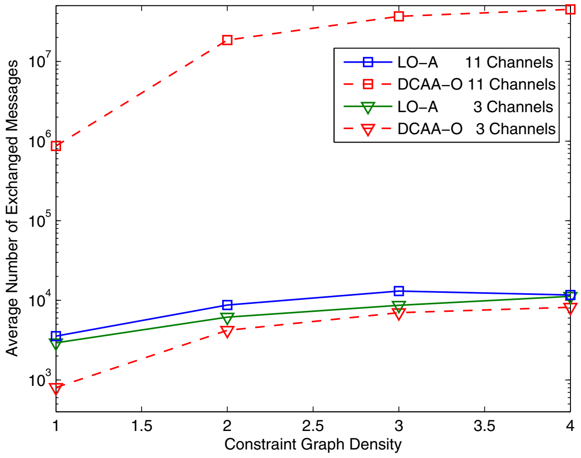

Fig. 3. Performance of DCAA-O and LO-A (9 APs)

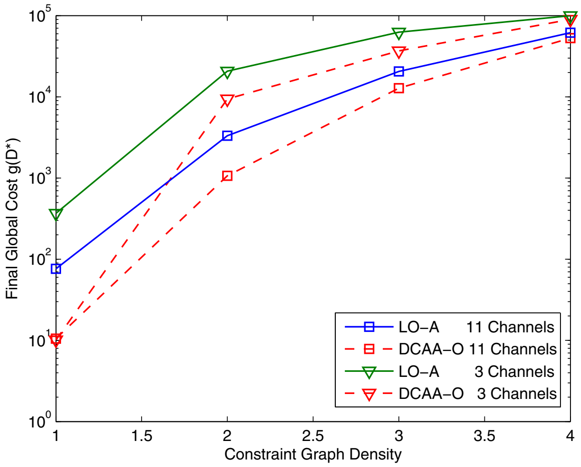

Fig. 4. Final global cost value for DCAA-O and LO-A (9 APs)

was the same and is omitted here for brevity. The performance evaluation of DCAA-O and LO-A for random topologies formed by 9 APs is shown in Figure 3. For the case of 3 available channels, DCAA-O presented lower values for the average number of exchanged messages compared to LO-A (around 50% less). However for the case of 11 channels, although reaching the optimal solution to the cost function, DCAA-O requires a larger number of exchanged messages for convergence than LO-A. As shown in Figure 4, it is important to note that DCAA-O always finds the optimal solution for every scenario, whereas LO-A algorithm achieves suboptimal values for graph densities 1, 2 and 3. For the case of 9 APs with density 4, the suboptimal values obtained by LO-A are close to optimal solution found by DCAA-O. The same conclusions also apply for scenarios with greater densities or with a larger number of nodes, as we observed in topologies with 16 APs. The results for 16 APs were omitted for brevity. It is important to point out that we also validated the DCAA-O solution optimality in each scenario by means of exhaustive search in the solution space. It is important to point out that the final global cost is directly related to the throughput performance metric because it measures the overall network interference. Therefore, by using the analytical cost model we do not need to run extensive throughput simulations.

7 Conclusions

The work reported in this paper combines artificial intelligence (AI) research methods with wireless network optimization problems, addressing the channel assignment in WLANs as a *distributed constraint optimization problem*. The proposed algorithm (DCAA-O) achieves the optimal solution for the channel assignment problem, therefore reducing the adjacent channel interference and maximizing the network performance. The required average number of exchanged messages among APs to reach optimal solution demonstrated to be inferior to LO-A for some specific scenarios. Scenarios with a larger number of APs and higher densities must be investigated in order to reduce the number of exchanged messages in DCAA-O.

Acknowledgments. This work was partially supported by CAPES (Brazil), under Grant RH-TVD: AUX-PE-RH-TVD 249/2008.

References

1. IEEE 802.11 Standard Group Web Site, `http://www.ieee.org/11/`
2. Aoun, B., Boutaba, R., Kenward, G.: Analysis of Capacity Improvements in Multi-Radio Wireless Mesh Network. In: Proceedings of IEEE VTC, vol. 2, pp. 543–547 (2006)
3. Ramachandran, K., Belding, E.M., Almeroth, K.C., Buddhikot, M.M.: Interference Aware Channel Assignment in Multi-Radio Wireless Mesh Networks. In: Proceedings IEEE INFOCOM 2007 (2007)
4. Chen, J.K., de Veciana, G., Rappaport, T.S.: Site-Specific Knowledge and Interference Measurement for Improving Frequency Allocations in Wireless Networks. IEEE Transactions on Vehicular Technology 58(5), 2366–2377 (2009)
5. Ko, B.J., Misra, V., Padhye, J., Rubenstein, D.: Distributed Channel Assignment in Multi-Radio 802.11 Mesh Networks. In: Proceedings of IEEE Wireless Communications and Networking Conference, WCNC 2007, pp. 3978–3983. IEEE, Los Alamitos (2007)
6. Li, K., Wang, F., Zhang, Y., Zhang, F., Xie, X.: Distributed Joint Resource Allocation in Multi-Radio Multi-Channel Wireless Mesh Networks. In: Proceedings of IEEE Global Telecomunications Conference (GLOBECOM), GLOBECOM 2009, pp. 1–6 (December 2009)

7. Leung, K.K., Kim, B.-J.: Frequency Assignment for IEEE 802.11 Wireless Networks. In: 58th IEEE Vehicular Technology Conference, VTC 2003-Fall, vol. 3, pp. 1422–1426 (October 2003)
8. Mishra, A., Banerjee, S., Arbaugh, W.: Weighted Coloring Based Channel Assignment in WLANs. In: ACM SIGMOBILE Mobile Computing and Communications Review (July 2005)
9. Mishra, A., Brik, V., Banerjee, S., Srinivasan, A., Arbaugh, W.: A Client-Driven Approach for Channel Management in Wireless LANs. In: IEEE Infocom 2006 (2006)
10. Leith, D.J., Clifford, P.: A Self-Managed Distributed Channel Selection Algorithm for WLANs. In: Proc. Int. Symp. Model. Optimization Mobile, Ad Hoc Wireless Net, pp. 1–9 (April 2006)
11. Modi, P.J., Shen, W.-M., Tambe, M., Yokoo, M.: An Asynchronous, Complete Method for General Distributed Constraint Optimization. In: Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems, Melbourne, Australia, pp. 161–168 (2003)
12. Freuder, E., Quinn, M.: Taking Advantage of Stable Sets of Variables in Constraint Satisfaction Problems. In: Proceedings of the International Joint Conference of AI, pp. 51–57 (1985)
13. Briggs, K., Tijmes, M.: Optimal Channel Allocation for Wireless Cities. In: VTC Spring (2009)
14. Yokoo, M.: Distributed Constraint Satisfaction: Fondation of Cooperation in Multi-agent Systems. Springer, Heidelberg (2001)
15. Wooldridge, M., Jennings, N.R.: Intelligent Agents Theory and Practice. Knowledge Engineering Review 10, 115–152 (1995)
16. Hirayama, K., Yokoo, M.: Distributed Partial Constraint Satisfaction Problem. Principles and Practice of Constraint, 222–236 (1997)
17. Gallager, R.G., Humblet, P.A., Spira, P.M.: A distributed algorithm for minimumweight spanning trees. ACM Transactions on Programming Languages and Systems 5, 66–77 (1983)
18. Gatani, L., Lo Re, G., Gagli, S.: An efficient distributed algorithm for generating multicast distribution trees. In: Proc. of the 34th ICPP - Workshop on Performance Evaluation of Networks for Parallel, Cluster and Grid Computer Systems, pp. 477–484 (2005)
19. Library SIMPATICA (2009), `http://www.ppgia.pucpr.br/~maziero/doku.php/software:simulation`