Uplink distributed resource allocation algorithm for maximized WLAN performance

Jean-Christophe Dunat and Christian Bonnet

***Abstract*— Despite the growing success of WLAN systems, current spectrum resource allocation algorithms used lack flexibility to closely follow the traffic variations and adapt the spectrum allocation in consequence, leading to performance loss.**

In this article we propose a flexible and powerful distributed spectrum allocation algorithm, called AB-DFSA, able to learn over time and with the objective to maximize the system throughput by using multi-user diversity, and able to quickly adapt the spectrum allocation in consequence. Such an algorithm finds its application in the scope of WLAN systems (e.g.: 802.11x).

***Keywords*— Distributed optimization, flexible spectrum allocation, opportunistic scheduling, WLAN, agent**

I. Introduction

Note that even if our study uses Sub-Carriers (SC)s as a spectrum resource, the presented algorithm is still applicable (with appropriate caution regarding inter-channel interference) to other types of systems. Also, we will refer to (1) "users" as "nodes", and (2) to "sum capacity" as the sum of the capacity per allocated SC over all the system SCs. The study presented hereafter finds its application in the context of spectrum pooling where an unlicensed spectrum band is shared by all the infrastructure equipments (Access Point (AP)s) using a wireless system (Wireless Local Area Network (WLAN)). Opportunistic methods to allocate spectrum resource are more than necessary to use, not to waste the precious spectrum resource and to closely and quickly adapt to the needs. If the spectrum allocation method takes into account multi-user diversity, the result is an increasing system sum capacity as the number of nodes increases [1].

In this article, the proposed algorithm called Agent-Based Distributed Flexible Spectrum Allocation (AB-DFSA) achieves a dynamic and flexible UpLink (UL) Orthogonal Frequency Division Multiple Access (OFDMA) SC allocation by exploiting multi-user diversity using a distributed approach with the objective to maximize the cell sum capacity. In addition, it has another very important advantage: its global parameters allow an operator to easily tune its network and change the optimization constraints to adapt its network Quality of Service (QoS). Such a multi-purpose algorithm is of particular interest to reduce the burden and cost associated with spectrum network planning, maintenance and upgrade.

The remainder of this paper is organized as follows. The optimization problem is described in Section II. Section III consists in a global overview of our distributed optimization algorithm. In Section IV we more precisely explain the heuristic used. In Section V we present some results obtained with this algorithm. Finally, we conclude the study in Section VI.

Jean-Christophe Dunat is in Motorola Labs, Parc Les Algorithmes, 91193 Gif sur Yvette, France (e-mail: jcdunat@motorola.com).

Christian Bonnet is in Institut Eurecom - BP 193, F 06904 Sophia-Antipolis, France (e-mail: christian.bonnet@eurecom.fr).

II. Optimization problem

First of all, let us define as "central node" the infrastructure equipment (AP), and as "nodes" the user equipments (Mobile Terminal (MT)).

The optimization problem considered can be described as follows (index i refers to the nodes and index j to the SCs). Given N available SCs to allocate among M nodes, each node i having a Signal to Noise Ratio (SNR) value γ_{ij} on SC j, find the best allocation of nodes on SCs that maximizes the cell sum capacity, subject to the following constraints: only a single node per SC at the end, and (n_i being the number of used SCs by node i) $n_i = 0$ (the current node i gets no SC) or $n_i \in [n_{min}; n_{max}]$ and $n_{max} < N$. The Shannon capacity achievable by node i on SC j (for a unit bandwidth) is simply: $c_{ij} = log_2(1+\gamma_{ij})$. We are interested in the continous case of coding and modulation. The objective is to find the final allocation vector for the system maximizing the system sum capacity, assuming all the constraints are respected. The central node broadcasts the range of n_i values. Then, each node internally regularly controls its n_i value to stay within the specified limits.

As this optimization problem is NP-complete, there exists no algorithm to find the optimal solution in all cases in a polynomial time. Accordingly, to find a "good" sub-optimal solution we used an heuristic.

III. Distributed optimization

This work is not a theoretical study on distributed optimization but rather it presents a flexible algorithm with an emphasis put on the feasability of its realisation for real systems.

The scenario envisaged consists of a single cell with a central node and several geographically distributed nodes in the cell. We assume a low mobility of the nodes, a known channel at the receiver, a symmetric channel with not "too rapid" variations over time, and SNR values do not change during the negociation part, and a synchronized UL reception of the messages.

We have assumed nodes have full responsibility in choosing their set of SCs to use for transmission, and our algorithm does not require the nodes to transmit all their information to the central node, but only the necessary information. We tried to reduce as much as possible the coordination overhead. Such a distributed approach is of particular interest when the number of attached nodes becomes large. Distributed optimization methods possess many advantages such as: scalability, robustness, adaptability to varying conditions. Also, distributed agents can be equiped with learning capabilites such as Reinforcement Learning (RL). Each agent runs the algorithm separately and by interacting with the other agents, they all contribute to create a globally optimal solution. As such, the number of tasks performed per node is reduced.

We used a distributed approach for UL allocation, with a feedback broadcasted by the central node. All nodes willing to participate in the SC allocation, simultaneously transmit an UL message to the central node. Thus, collisions are encouraged and handled during the negociation phase, in order not to have one during the data transmission. This way, the problem of hidden nodes is completely removed and there is no risk of performance degradation due to collisions during the data transmission phase. Our algorithm is scalable with the number of nodes and able, without any prior knowledge of the number of nodes, to dynamically adapt to the following 3 scenarios: $N = M$, $N < M$ and $N > M$, as well as to any values of n_{min} and n_{max}.

We propose a method consisting in a succession of short UL and DownLink (DL) messages to coordinate the distributed allocation. Nodes do not communicate with each other, but only require to exchange information with the central node to coordinate their allocation. Then, they receive the result by central node broadcast. The central node only acts as a filter extracting the maximum and the minimum values out of all the received values, and sending it back to the nodes for information sharing. To reduce the coordination overhead between all the nodes, we used a parallel exchange of information.

The technique we used consists in an UL simultaneous exchange of information by all active nodes towards the central node. By forcing all contenders (even hidden nodes) to express their desire at the same time and by broadcasting a result (the maximum value and the minimum value per SC) we were able to completely remove the hidden node problem! The capacity value c_{ij} is time-coded. Practically, each node will code its capacity value and transform it into a message duration such that the shorter the message, the greater the capacity.

In practice, it is very easy for the central node to detect the maximum capacity value: by detecting the first drop (simple gap detection rule) in the aggregation of the received messages. Duration between the start and the first drop gives the corresponding transmitted value. By detecting the total duration of the longer message the central node knows the minimum capacity value. As the optimization progresses, only the best values will be kept. The DL message will be sent only after the end of the longest SC message. An advantage of this method is the simplicity of the coding and decoding, robustness against interference and collisions, and it does not require an orthogonal coding for each node or each capacity value.

At the end of the optimization phase, a given SC is allocated to only a single node, but a node can be specialized in several SCs. During the optimization process there is both an inter-node negociation (to respect the unicity of node allocation per SC) and intra-node negociation (due to the n_i constraints). Indeed, each node would want to use all the n_{max} SCs, but this resource must be optimally shared among all the nodes to globally maximize the cell UL sum capacity, by means of local interactions.

After the end of the SC allocation phase, the allocated nodes all simultaneously transmit their data frame (same and fixed duration over all nodes), followed by an acknowledgment frame broadcasted by the central node. Then, a new SC allocation phase takes place, and so on.

IV. Modelling and description of the heuristic

To solve the considered optimization problem we used a general heuristic inspired from a model of division of labor used by social insects such as ants, wasps, etc. This heuristic comes from the domain of biology (studies of social insects) then modeled by the domain of artificial intelligence, improved and successfully applied to solve real world optimization problems, using a set of cooperating agents. This heurisitic was proven to be very efficient to optimize real life problems of dynamic and distributed tasks allocation in the industry [2], [3].

Agents use the same finite set of simple rules to interact with each other. The result of this cooperation is an auto-organization at the system level. The modelling challenge is to find the appropriate set of local rules that will globally create the desired behavior. We managed to successfully adapt and use this heuristic to our problem with good results, thus showing its great potential for optimization and control of cooperating agents.

The ability of an agent to perform a task depends on its ability for this task, as well as the number of other agents willing to perform this task. Thus, agents "specialize" over time in the tasks where they are best fitted for, by means of a variable threshold as well as by inter-agent conflicts (dominance contests).

The variable threshold with time of use model [4] is able to appropriately adapt to very dynamic environments. The more an agent becomes specialized in a given type of task, the greater the probability to engage in performing it in the future. Each agent maintains and update over time a threshold per type of job to be done.

In our model, a node contains (1) a constraint controller and (2) a set of N cooperating agents (one by SC). The constraint controller ensures that the contraints in number of active SCs per node are always ensured. The main objective of an agent is to stay active (ON) at the end of the optimization process. Agents interact internally (with other SC-agents within the same node) and externally (with agents representing the same SC index but from other nodes). The rules of interactions of agents between them, are build to reach a global optimum by using local interactions, as in auto-organisation. While they interact, agents exchange information about capacity values.

To become stronger and have better chances to stay active, agents organize in groups (within each node) and try to find the best other members of their group for them to be ON at the end. The weakest agents are left apart and groups are reconstructed until the end of the optimization process. Groups are organized and re-organized dynamically and autonomously to reach the best final configuration for the system.

We assume that a node is able to simultaneously perform n_{max} tasks out of a total of N tasks. An agent is able to perform only a single task, and it can either perform it (ON status for that SC) or not.

In our model, an agent's threshold value changes over time and its value depends on (1) its previous history, and (2) its ability to perform its task better than the agents from other nodes for that same task (i.e. SC). The threshold value will modify the probability of that agent to perform its task in the future, such that: a low threshold leads to a high probability to perform this task, whereas a high threshold means a low probability to perform this task.

In optimization, there is an important trade-off which is to equilibrate between exploitation of the current solution and exploration of new solutions. The chosen heurisitic, owing to its variable threshold model, adapts well to a dynamic environment and leads to faster convergence (but not premature) than many other heuristics. It is also very efficient in avoiding local optima. It uses a simple RL algorithm described hereafter. In addition, our model is able to adapt to a large set of constraints. The result is that, with the same algorithm, we are able to solve the problem of SC allocation for a range of SCs per node from 1 up to N. The particular case where a single node can capture all N available SCs can be interpreted as a new allocation method for a single channel system.

The first time, each node starts with its n_{max} best SCs ON, before transmitting in UL. Then, the following steps (1) to (4) are repeated until all constraints are ensured : (1) the amplitude of the threshold variation $\Delta\theta_{ij}$ is determined (according to a given function) per SC and per node, (2) each node updates all its threshold values θ_{ij}, (3) then each agent i updates its status per SC (ON or OFF) by evaluating the probability p_{ij} (function of its threshold θ_{ij}) to perform task j using a Fermi-Dirac distribution. Finally, (4) each node's constraint controller must control if n_i is within the acceptable values.

Our algorithm contains several global and easy to control parameters, allowing to tune the quality of the solution and the speed of convergence as needed.

V. Some results

The numerical values used in the following scenario is modified values taken from an IEEE 802.11a system. However, other values could be taken without altering the conclusions.

Taking as an example an 802.11a system [5] in the band $[5.25-5.35]$ GHz, we calculated the maximum range of SNR values for any cell node. The maximum range of capacity values per SC was deduced. After calculations and taking into account the maximum transmit power regulations, the SC bandwidth and the set of possible transmit modes, we obtain that all SNR values γ_{ij} $(\forall i, j)$ are within: $[\gamma_{min};\gamma_{max}] = [13.5;78.07]$ dB . In our study, only the continuous capacity case is considered. We can now transform these SNR values to find out the maximum capacity range c_{ij} $(\forall i, j)$ are within: $[c_{min};c_{max}] = [4.54;25.9]$ b/s/Hz (for a SC bandwidth of 1 Hz). c_{ij} values are uniformly drawn within $[c_{min};c_{max}]$. Note, that the aim of this study was not to focus on the channel model, but rather to demonstrate the benefit of the proposed algorithm.

The results presented were obtained for the following range of values: $N = 10:10:50$, $M = 10:10:100$, $n_{min} = 1$ and $n_{max} = 0.1 \times N$. In other words, each node is allowed to be allocated up to 10% of the total spectrum resource. The n_{min} and n_{max} constraints can be interpreted as setting limits of total capacity per node, assuming we know the distribution of capacity values per SC. Setting such limits for nodes allow to control that not a single node will be allocated the entire spectrum resource, as well as that there is at least a minimum bandwidth per node to run a given service.

Each point in the following figures corresponds to a mean over 10 different and randomly selected new set of values.

The maximum total sum capacity corresponds to the allocation of N values at c_{max}, which is where the system tends for an infinite number of nodes when using multi-user diversity. As seen in Figure 1, the total sum capacity

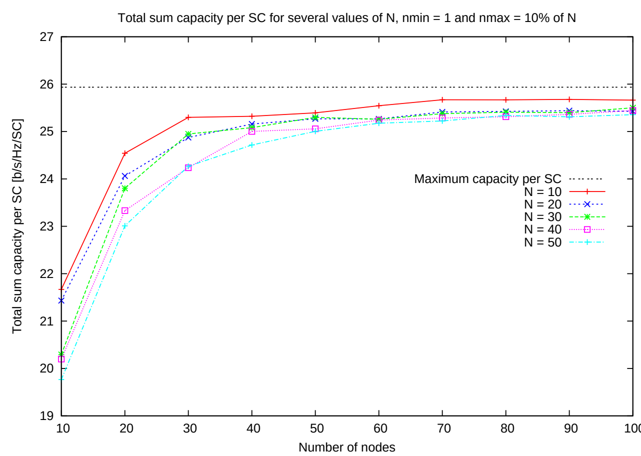

Fig. 1. Total sum capacity per SC for varying N and M, nmin = 1 and nmax = 10% of N

per SC increases with the number of nodes M and tends towards the maximum total sum capacity per SC, due to the multi-user diversity (nodes with better capacities can be chosen). Note that as $n_{min} = 1$, the resulting cell allocation contains no idle SC, they are all allocated. Also, the system sum capacity increases with N but slower than the N factor, which explains why the sum capacity per SC is reduced while the number N of available SCs increases. The reason is that as $n_{max} = 0.1 \times N$, the SC block size increases, leading to less and less exploitation of the multi-user diversity.

After discussing the quality of the obtained solution, Figure 2 presents the speed of convergence to it, in terms of mean number of iterations per SC. As seen in Figure 2, the mean number of iterations per SC to reach the final cell allocation is reduced as M increases. In other words, for a given N, it does not take longer to converge with an additional number of nodes in the cell. This proves the scalability in terms of nodes of the algorithm. In addition, there is an increase in the total mean number of iterations but which is smaller than the N factor, for the same reasons than before.

Accordingly, for an increasing problem size (N increases), the smaller the convergence time per SC, the

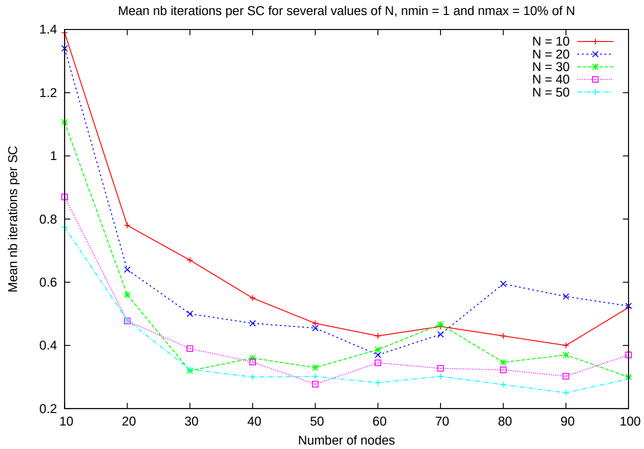

Fig. 2. Mean number of iterations per SC for varying N and M, nmin = 1 and nmax = 10% of N

smaller the capacity per SC. Note that the n_i constraints have an impact on the quality of the solution and the number of iterations. In other words, there is a tradeoff to consider between speed of convergence towards the final solution and its quality.

As each node is allowed to be allocated n_i SCs with $n_i \in [1 : 0.1 \times N]$, we also investigated the resulting mean number of SC per allocated node (refer to Figure 3). The

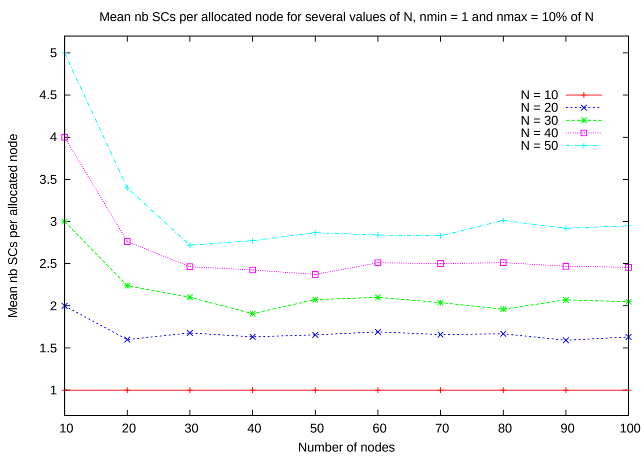

Fig. 3. Mean number of SC per allocated node for varying N and M, nmin = 1 and nmax = 10% of N

mean number of SC per allocated node quickly stabilizes as M increases. We can verify on Figure 3 that the mean number of SC per allocated node is approximately equal to $\frac{n_{max}+n_{min}}{2}$. Some nodes will be allocated more SCs than other nodes. In other words, each node is able to autonomously adapt its number of contending SCs within the authorized limits, in order to reach no overlap at the cell level at the end of the optimization phase, while trying to maximize its own total capacity.

The n_i contraints can be tuned to allow the coexistence of various services in the same cell, each with its own n_i limits. The algorithm would be able, without the need to know the penetration rate of each service, to autonomously accomodate the best set of spectrum resource to each node, given its service requirements.

VI. Conclusions

The distributed algorithm presented in this article is very promising in the context of flexible spectrum allocation. The obtained results are very good and the algorithm can be easily controlled by an operator to change the optimization constraints according to its proposed services, and its network characteristics. In addition, the algorithm is applicable for a large range of optimization constraints even though the problem in itself is very complex. The operator is able to control all the algorithm parameters in order to choose between number of iterations to reach the cell allocation or quality of the cell sum capacity. This can be motivated to adapt to several conditions of channel speed and amplitude of variation. We believe this algorithm will be very efficient for next generations of WLAN systems, to opportunistically use the appropriate part of the spectrum according to the nodes' needs.

References

[1] R. Knopp "Achieving Multiuser diversity under Hard Fairness Constraints", *Proceedings. 2002 International Symposium on Information Theory*, 2002, pp. 451.

[2] E. Bonabeau, G. Theraulaz and J.L. Deneubourg "Adaptive task allocation inspired by a model of division of labor in social insects", *In D. Lundh and B. Olsson, editors, Bio Computation and Emergent Computing*, pp.36-45, World Scientific, 1997.

[3] V.A. Cicirello and S.F. Smith "Improved routing wasps for distributed factory control", *In IJCAI-01 Workshop on Artificial Intelligence and Manufacturing: New AI Paradigms for Manufacturing*, August 2001.

[4] V.A. Cicirello and S.F. Smith "Distributed Coordination Resources via Wasp-like Agents", *In WRAC*, January 2002.

[5] "IEEE 802.11a. High Speed Physical Layer in the 5 GHz band. Supplement to Standard IEEE 802.11", *IEEE*, New York, 1999.