Hindawi Publishing Corporation
Journal of Electrical and Computer Engineering
Volume 2010, Article ID 402831, 10 pages
doi:10.1155/2010/402831

Research Article

Distributed Graph Coloring for Self-Organization in LTE Networks

Furqan Ahmed,[1] Olav Tirkkonen,[1,2] Matti Peltomäki,[3] Juha-Matti Koljonen,[3] Chia-Hao Yu,[1] and Mikko Alava[3]

[1] *Department of Communications and Networking, Aalto University, P.O. Box 13000, 00076 Aalto, Finland*
[2] *Nokia Group, Nokia Research Center, P.O. Box 407, 00045 Helsinki, Finland*
[3] *Department of Applied Physics, Aalto University, P.O. Box 14100, 00076 Aalto, Finland*

Correspondence should be addressed to Olav Tirkkonen, olav.tirkkonen@tkk.fi

Received 1 April 2010; Revised 1 July 2010; Accepted 27 August 2010

Academic Editor: Seppo Hämäläinen

Primary Component Carrier Selection and Physical Cell ID Assignment are two important self-configuration problems pertinent to LTE-Advanced. In this work, we investigate the possibility to solve these problems in a distributive manner using a graph coloring approach. Algorithms based on real-valued interference pricing of conflicts converge rapidly to a local optimum, whereas algorithms with binary interference pricing have a chance to find a global optimum. We apply both local search algorithms and complete algorithms such as Asynchronous Weak-Commitment Search. For system level performance evaluation, a picocellular scenario is considered, with indoor base stations in office houses placed in a Manhattan grid. We investigate a growing network, where neighbor cell lists are generated using practical measurement and reporting models. Distributed selection of conflict-free primary component carriers is shown to converge with 5 or more component carriers, while distributed assignment of confusion-free physical cell IDs is shown to converge with less than 15 IDs. The results reveal that the use of binary pricing of interference with an attempt to find a global optimum outperforms real-valued pricing.

1. Introduction

Self-organization is a wide ranging research and standardization trend in modern networking. In the scope of wireless networking, research on Self-Organized Networks (SONs) ranges from general principles of cognitive and ad hoc network to concrete problems in standardization and implementation of near future mobile networks [1, 2]. Here, we concentrate on a specific SON problem of current interest for the standardization of the next release of the Evolved Universal Terrestrial Radio Access Network (E-UTRAN), a.k.a. Long Term Evolution (LTE). This release, being standardized by the 3rd Generation Partnership Project (3GPP), is known as LTE-Advanced (LTE-A), being standardized by the 3rd Generation Partnership Project (3GPP).

New features of next-generation wireless networks will have an impact on SON, resulting in new use cases and requirements. Of particular interest are Local Area deployments of femto-and picocells. One of the four evaluation scenarios for IMT-Advanced is an indoor scenario [3]. This lends relevance to studying not only automated networking functions, but also autonomous functions. In this paper, we concentrate on two autonomous self-configuration functions, which can be mapped to a graph coloring problem.

The first problem we address is Primary Component Carrier Selection. In 3GPP discussions, carrier aggregation is an essential feature of LTE-A [4]. This leads to the problem of component carrier selection—an individual Base Station (BS) may potentially not operate on all the aggregated carriers, but just on a subset of them. A viable alternative for robust operation is that each BS selects one carrier as a primary carrier, on which the BS has a full set of control channels with full coverage [5, 6]. The Primary Component Carrier Selection (PCCS) problem as such is a direct relative to the well-studied frequency assignment problem [7]. In [5, 6], an autonomous version PCCS was discussed. Another use case for self-configuration, discussed in [1, 8], is automated Physical Cell ID (PCI) assignment. In LTE, the physical cell

ID is needed to distinguish the signal of one BS from the signal of another. Accordingly, neighboring BSs should not have the same PCI. Moreover, to avoid confusion in Hand-Over (HO), a cell should not have two neighbors with the same PCI. Thus the PCI configuration problem becomes a graph coloring problem on the graph of two-hop neighbors. Graph coloring aspects of PCI assignment in LTE have been addressed in [9], where a centralized approach was discussed. A distributed solution based on reserving part of the ID-space to be used for newly switched on cells, was considered in [8]. Apart from [8], to the best of our knowledge, the problem has not been addressed in a distributed manner before.

We investigate simple distributed graph coloring algorithms for both applications addressed. Performance is analyzed for indoor environments based on office houses of the type discussed in [10], placed in a Manhattan grid. A realistic model on UE measurements and reporting is used. Neighbor relations between BSs are determined by handover measurements performed by the User Equipments (UEs). We investigate a dynamical network setting, where the network grows by adding BSs, one by one. When a new BS is added, it self-configures, based on measurements and discussions with the neighbors. We observe that when the measurement and reporting load of the UEs is small, it is beneficial to base distributed decisions on binary conflicts, not real interference couplings. Also, we find the minimum number of component carriers and physical cell IDs that are required for these distributed algorithms to converge.

This paper is organized as follows. In Section 2, we discuss the self-configuration problems addressed. In Section 3, we discuss simple distributed graph coloring algorithms, and their properties. In Section 4, system and network models are presented. Section 5 discusses results for Primary Component Carrier Selection, and Section 6 is on Physical Cell ID Assignment. Finally, conclusions are given in Section 7.

2. Self-Configuration Problems and Coloring

2.1. Autonomous Primary Component Carrier Selection. In carrier aggregation, an operator aggregates a number of component carriers for LTE-A operation. The component carriers may have any of the allowed LTE bandwidths (1.4, 3, 5, 10, 15, or 20 MHz), and they may be either contiguous or noncontiguous [4]. It is natural to take one of the component carriers as a primary one in each cell [5, 6]. This primary carrier serves mobility purposes, and has accordingly a full set of control channels with maximum coverage. The other carriers may be used to boost the data rate, when needed. From the perspective of guaranteeing control channel coverage, the selection of the primary component carrier becomes a classical frequency assignment problem (FAP) [7]. When no other issues than interference is taken into account, and each resource is indistinguishable (i.e., there are no specific channel-related reasons for a BS to favor one carrier more than another), FAP is equivalent to graph coloring. However, when the problem is addressed in an autonomous manner, such as in [6], the local decision makers (BSs) may have more local information at hand for making the decision. Thus the coloring problem may be addressed based on real-valued interference costs, not just on conflicts, as in the classical approaches. For example, in [6], each BS selects the primary component carrier according to real-valued interference information of the neighbors.

Here, we address Primary Component Carrier Selection as a distributed graph coloring problem. According to the discussion in [4], the relevant number of colors for this case is less than ten.

2.2. Autonomous Physical Cell ID Assignment. In LTE, the physical cell ID N_{ID} determines the structure of many channels used in the cell. The ID itself is given by the synchronization channel, and there are 504 different ones. An important use of PCI is to separate cells in handover (HO) measurements. To guarantee proper cell search and handover performance, the PCI assignments should be:

(i) conflict-free: the PCI should be unique in the cell area—no neighbors that the UEs may synchronize to and consider as HO candidates should have the same PCI;

(ii) confusion-free: a cell should not have two neighbors with the same PCI—this guarantees that outward handovers are treated in a proper manner.

Conflict freeness makes the PCI assignment problem a graph coloring problem on the graph of neighbors. Confusion freeness makes it a graph coloring problem on the graph of two-hop neighbors. As any two neighbors of a cell should not have the same PCI, this means that no cell should have a two-hop neighbor (a neighbor of a neighbor) with the same PCI [9]. The space of PCIs may be divided into smaller parts for multiple reasons. For example, parts of the PCI space may be reserved for different layers (macro/micro/femto layers), or part of the space may be restricted for newly switched on, or reconfiguring BSs [8]. In addition, the PCI explicitly determines the structure of the downlink (DL) and uplink (UL) reference signals [11], and this may lead to a much tighter problem for PCI assignment. Related to downlink, there are six different subcarrier groups that downlink reference signals may be mapped to. The subcarrier shift is determined by N_{ID} mod 6. In normal shared channel operation there may not be significant differences related to which subcarriers neighboring BSs have their reference signals on. However, if Collaborative Multipoint transmission (CoMP) with joint beamforming [4] is employed, the reference signal placement becomes an issue. In joint beamforming CoMP, a UE may receive a joint transmission from multiple BSs. In this case, it should be capable of reliable estimation of the channel from these BSs. This requires reference signals to be orthogonal. If orthogonality is achieved in the frequency domain, as in LTE, neighboring BSs should have different subcarrier shifts, and accordingly different N_{ID} mod 6.

The uplink reference signals are grouped into 30 sequence groups, so that sequences with most severe cross-correlations are grouped into the same group. It is desirable

that neighboring cells use different sequence groups, so that it can be guaranteed that reference signals in neighboring cells have good cross-correlation properties. In addition, group hopping is possible. The group used by a cell at any time is determined by $N_{\mathrm{ID}} \bmod 30$, and accordingly, neighboring cells should have differing $N_{\mathrm{ID}} \bmod 30$. Based on this, we observe that PCI assignment in LTE may require conflict freeness with 30 colors, and with CoMP, conflict freeness with 6 colors. For confusion freeness, a larger space of colors is possible. It is worthwhile to investigate, how many PCIs are needed to guarantee confusion-free autonomous PCI assignment.

3. Distributed Graph Coloring Algorithms

Much of the previous work on distributed graph coloring algorithms concentrates on finding colorings with $\Delta + 1$ or $O(\Delta)$ colors, where Δ is the largest number of neighbors of any node. For these cases, rapidly converging distributed algorithms exist, both deterministic and stochastic, and the convergence characteristics can be analyzed in closed form; see [12] and references therein. For more greedy cases, when the number of colors is smaller than Δ, generic constraint satisfaction algorithms may be used. In the survey paper [13], three algorithms proposed by Yokoo and Hirayama are discussed. Distributed Breakout (DBO) is an algorithm based on local reasoning that is capable of breaking out from a local minimum. Asynchronous Backtracking (ABT) and Asynchronous Weak-Commitment Search (AWC) are complete algorithms that are able to satisfy all constraints if possible. In [14], Distributed Stochastic Algorithms (DSAs) are considered, and shown to outperform DBO. DSAs are synchronous algorithms belonging to the wide class of local search algorithms; see [15] for a review.

We want to perform the allocation of the resources so that the BSs execute a routine asynchronously, that is, one at a time, and the order in which they do this is random but fixed. This models an operation where each BS updates its decision at regular intervals, according to a clock which is not synchronized with its neighbors. Also, the time it takes to communicate the change to the neighbors is assumed to be negligible compared to the update interval. Accordingly, we will select a few asynchronous local search algorithms, as well as AWC and ABT, for evaluation.

3.1. Local Search Algorithms. In this paper, four simple distributed local search algorithms will be used for graph coloring. The considered algorithms can be classified according to two characteristics.

The first classification is related to the type of interference pricing. When real-valued interference couplings are used, a real-valued price may be considered between neighbors using the same resource [16, 17]. In contrast, when binary conflicts are considered, the decisions are made based on the number of conflicting neighbors only, not based on the strength of the conflicts.

A second classification is according the number of alternatives tried by a node, when updating which resource to use. The first alternative is a random selection of resource, as in Monte Carlo algorithms; see [18]. The BS randomly selects one of the resources not used by itself at the moment. It starts to use the new resource, if the price (interference price, or number of conflicts), is less than or equal to the price with the earlier resource. In a multiple-try algorithm (e.g., [14]), the node calculates the price for all resources, and randomly selects one of the resources with the lowest price. The binary algorithms have an *Absorbing Local Optimum*, that is, the BS does not change its resource once it is in a conflict-free state [14, 19]. The motivation for this is that in a global optimum of a graph coloring problem, each node sees a local optimum. With this method, unnecessary reconfigurations of the BSs are avoided. In addition, when a node is not in a local optimum, we allow *plateau moves*. This means that if the price of the tested resource, or with multiple-try, the lowest price of all resources, is the same as the price of the resource being used, the node changes the resource. This allows the algorithm to break out from a local optimum and search for a global optimum, as discussed in [14, 15].

The resulting four algorithms are called

(i) Bin: binary pricing, random candidate resource selection,

(ii) Real: real pricing, random candidate resource selection,

(iii) BinMulti: binary pricing, all candidate resources considered, random selection among the best candidates,

(iv) RealMulti: real pricing, all candidate resources considered, random selection among the best candidates.

The real-pricing algorithms with random, and best candidate selection are known from [16, 17], respectively. The binary multiple-try algorithm is an asynchronous version of DSA-D of [14] with $p = 1$. The simplest algorithm, binary pricing with random candidate selection, is to our knowledge not known in the literature as a distributed algorithm. Similar principles are used when solving problems in statistical physics using the so-called zero-temperature Markov Chain Monte Carlo algorithm [18]. The four local search algorithms can be compactly described as follows. Let $G(\mathcal{V}, \mathcal{E})$ be a graph where the BSs are the vertices $v \in \mathcal{V}$, $\mathcal{E}$ is the set of edges, and $w(v, v')$ is the weight of an edge connecting v and v'. For real-valued pricing, $w \in \mathbb{R}$, whereas for binary pricing, $w \in \{0, 1\}$. The set of neighbors of v is $\mathcal{N}_v = \{v' \mid w(v, v') > 0\}$, and the set of colors is $\mathcal{C}$. If v uses color c_v, the local interference price experienced by v is

$$P_v(c_v) = \sum_{v' \in \mathcal{N}_v} \delta(c_v, c_{v'}) w(v, v'), \tag{1}$$

where δ is the Kronecker delta symbol. When v starts executing the routine, its color is c_v. The candidate new color is

$$\tilde{c}_v = \begin{cases} \mathrm{rand}(\mathcal{C} \neg \{c_v\}), & \text{for random selection,} \\ \mathrm{rand} \arg\min_{c \in \mathcal{C}} P_v(c), & \text{for multiple try,} \end{cases} \tag{2}$$

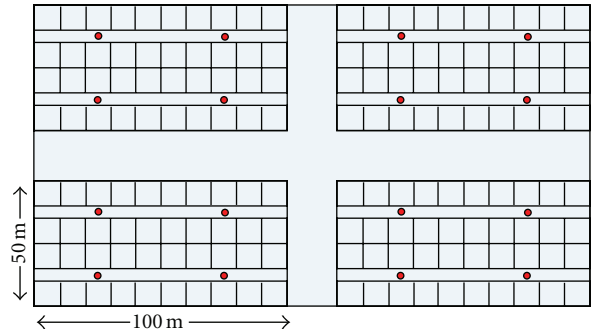

Figure 1: Layout of four-building scenario in the WINNER 2 office Manhattan path loss model. Red dots represent pico-cell BSs.

where rand($\mathcal{X}$) selects a random element from the set $\mathcal{X}$. The color after execution is

$$\begin{cases} \widetilde{c}_v, & \text{if } P_v(\widetilde{c}_v) \leq P_v\,(c_v),\ P_v(c_v) > 0, \\ c_v, & \text{otherwise.} \end{cases} \tag{3}$$

3.2. Complete Constraint Satisfaction Algorithms. For comparison, we consider two complete constraint satisfaction algorithms discussed in [13]. Both utilize global IDs to break the symmetry between agents, and operate in principle on a complete search tree. In Asynchronous Backtracking, priority values of agents follow global IDs and agents communicate their current values to neighbors asynchronously, using messages. Preceding agents in alphabetical order have higher priority. Agents try to find value assignments consistent with higher priority agents. If no such value exists, new constraints are generated and communicated to a higher priority agent, which then attempts to change its assignment. A value once selected is not changed unless lower priority agents force it. This makes a wrong value selection very expensive for large scale problems.

Asynchronous Weak-Commitment Search is based on a message passing principle similar to ABT. AWC improves on ABT by making use of a minimum conflict heuristic and dynamic reordering of agents (i.e., changing priorities) to minimize the number of constraints generated. This often enables recovery from bad value selection without an exhaustive search.

In ABT and AWC, a constraint received from a lower priority agent may involve the state of an agent that is not a neighbor of the receiving agent. To handle this, ABT and AWC require a protocol of establishing new communication links between agents, so that information pertinent to constraints is available at the agents. For more details on these algorithms, see [13].

For larger networks, a preprocessing step of graph partitioning may be used first to divide the graph into a set of smaller loosely connected ones, hence, enabling an efficient framework for obtaining the solution to underlying smaller Constraint Satisfaction Problems in a concurrent way. This approach is investigated in [20].

Table 1: Noise level parameters.

Transmit power P	20 dBm
Noise figure	9 dB
Signal bandwidth	20 MHz
BW efficiency	0.9

4. System Model

For performance analysis of the discussed algorithms for both Primary Component Carrier Selection and automated PCI configuration, in a picocellular network, an office building path loss model in a Manhattan grid has been constructed. The nodes correspond to base stations or cells, and the edges are weighted by real or binary-valued interferences. The path loss model and operation of BSs and UEs is discussed in following subsections.

4.1. Path Loss Model. The main propagation characteristics are according to the Winner path loss models in [10]. We consider both intra- and interbuilding interference by placing a number of multiple-floor buildings in a Manhattan grid; see Figure 1. The buildings model modern office buildings comprising of rooms and corridors. Propagation inside the buildings is modeled according to the Winner A1 model of [10], and propagation between the buildings is modeled as the Manhattan-grid path loss model B1 of [10]. Distance dependent path loss is calculated from the parameters A, B, C as

$$PL = A\log_{10}(d) + B + C\log_{10}\left(\frac{f_c}{5}\right) + X + FL, \tag{4}$$

where d is the distance between the transmitter and receiver, f_c is the carrier frequency, X is the wall and window loss, and FL is the floor loss.

As we are interested in the performance of a large system, we have chosen the buildings to have many floors. Also, we consider wrap-around boundary conditions in all directions (including the floor-dimension). The modeled system thus consists of a Manhattan grid of very tall buildings, and is essentially three-dimensional.

In addition to distance-dependent path loss, we consider shadow fading, according to [10]. It is assumed that the measurements UEs are performing are averaged over the channel coherence time. Accordingly, no fast fading is modeled. Parameters determining the thermal noise level can be found in Table 1. The parameters model a picocellular transmitter utilizing a full LTE bandwidth.

4.2. Model of UE Measurements and Reporting. To have a realistic model of the information that the BSs base their decisions on, we assume that the UE selects the strongest BS to be the serving BS, and reports some of the strongest interferers to the serving BS. If the measurement capabilities of the UEs are taken into account, not all UEs will be able to measure all BSs. We model this with a *synchronization threshold* H_{synch}, which is a threshold in SINR under which

Table 2: Neighbor relation parameters.

Synch threshold	$H_{\text{synch}} = -7$ dB
Reporting load	$L_{\text{rep}} = 1$
UEs per room	1
UEs per corridor	5
Cell selection	best C/I

a UE is not able to synchronize to a BS. This threshold is determined by the synchronization sequences used in the standard, and UE implementation. For LTE, a typical value would be -7 dB. The measurement and reporting load of the UE is taken into account by limiting the number of neighboring BSs that the UE should measure and report to the serving BS to a small number L_{rep}.

4.3. Model of BS Operation. We model the situation where the BSs collect information from their area for a sufficient time, before they start changing the network configuration. We consider the spatial coherence properties of shadow fading to be such that shadow fading is constant within a room. Thus we consider that sufficient statistics of the network is collected when there is one sample per independent shadow fading realization, meaning one UE per room. To consider a constant UE density, we assume 5 UEs per corridor. The parameters determining the neighbor relations are summarized in Table 2.

Once the BS has gathered sufficient information, it decides a cost for the interference caused by another BS to the UEs it serves. For each interferer, the BS has statistics of the interference produced, according to the received reports. In our model, the BS simply considers the worst interference caused by the interferer to any of its served UEs as the interference cost caused by that interferer. The interferences caused to different UEs are measured in terms of the Carrier-to-Interference ratio measured when synchronized to the interferer—the signal power from the interferer divided by the signal power from all other BSs (including the serving one).

4.4. Neighbor Relation and Interference Coupling. The neighbor relation between the cells is determined on a per-drop basis. Each drop represents a network configuration with fixed shadow fading, and a fixed neighbor relation. These relations are determined per drop as follows.

(i) 50 UEs are dropped per floor, evenly distributed in rooms and corridors.

(ii) UEs perform cell selection; serving cell is cell with best C/I.

(iii) Each UE selects primary HO candidate, BS with second best C/I.

(iv) If C/I of primary HO candidate is above the synchronization threshold H_{synch}, synchronization to primary HO candidate is considered successful. Otherwise the UE has no HO candidate.

(v) All BSs that are primary HO candidates of a UE served by the BS, are considered neighbors of the BS.

(vi) The interference coupling between a cell and its neighbor is the highest interference (relative to the carrier power) caused by this neighbor to a UE served by the cell. The interference coupling of non-neighboring cells is 0.

The resulting distribution of the number of neighbors per cell arising from the used neighbor definition can be found in Figure 2(a). For component carrier selection, and PCI conflict freeness, the number of neighbors is essential. For PCI confusion freeness, a BS should have no two-hop neighbors with the same ID, accordingly, the distribution of two-hop neighbors is relevant. This is depicted in Figure 2(b). The distributions are collected over 500 drops. It should be noted that in the simulated scenario, there are 96 BSs, and in one building, there are 24 BSs. The median number of neighbors is 3, and the median number of two-hop neighbors is 9.

4.5. Model of Network Growth. Performance is estimated in a growing network. First a randomly selected subset of 86 of the total 96 BSs in the network scenario are switched on. These are colored with a carrier/PCI according to the problem investigated in a conflict and, for PCI, confusion-free manner. The remaining 10 BSs in the network are then switched on, one-by-one. When a BS is switched on, it first selects a color, and then starts to execute a distributed carrier/PCI selection algorithm.

5. Graph Coloring for Primary Component Carrier Selection

In this section, we compare the performance of network algorithms employing different distributed graph coloring methods for Primary Component Carrier Selection. The idea is to share a small number of resources, which represents a component carrier that a LTE-A system may use, as efficiently as possible. Each cell selects a primary carrier. The output of the algorithm is measured by the distribution of SINR experienced at the nodes, once the primary carrier is distributed. The traffic model is static, and for simplicity it is assumed that there is no secondary usage of the resources. The aim is to analyze the characteristics of the different classes of graph coloring algorithms discussed above, especially related to their local versus global optimization characteristics. The performance metrics evaluated are the following.

(i) Probability of convergence and number of iterations: whether or not the network will be able to find a conflict-free state. In such a state, there are no conflicts above H_{synch} (real-valued or binary).

(ii) Number of cell reboots per added BS: when BSs running a routine of a distributed network algorithm are able to find a conflict-free primary component carrier configuration, the number of times any BS has changed its carrier in the process is measured. This should be a small number, preferably less than one.

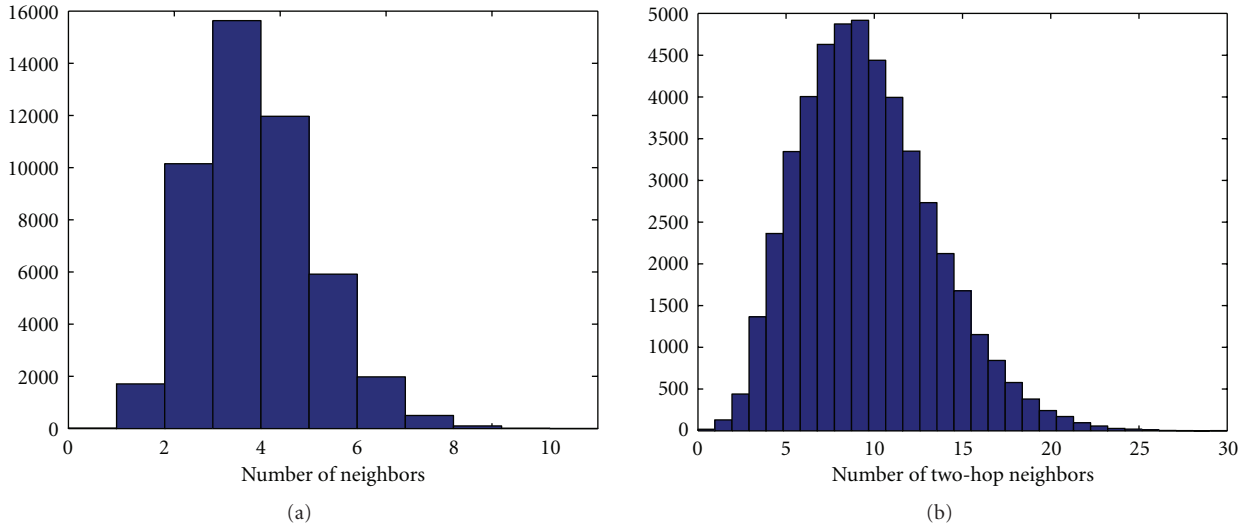

Figure 2: The distribution of the number of neighbors (a) and the number of two-hop neighbors (b).

(iii) The resulting Carrier-to-Interference (C/I) ratios of the selected primary component carriers. This is the most important measure in this case.

The algorithms used are the local search algorithms discussed in Section 3, two alternatives related to pricing, real-valued or binary, and two alternatives related to selecting a new resource to try: random or multiple try.

After switching on, a BS randomly selects a primary component carrier, starts serving UEs and collecting HO measurements. Once this is done, it starts executing a distributed graph coloring algorithm. The algorithm is run until it converges (to a local optimum for real-valued pricing, or to a global optimum for binary pricing), or until 1000 iterations have been done. An iteration is a cycle during which all BSs try to update the resource used once. After that, statistics of the experienced Carrier-to-Interference (C/I) ratios experienced by the users in the system are gathered. Based on these statistics, metrics for comparing system performance can be evaluated.

In Figure 3, the convergence properties are plotted. It can be seen that the binary algorithms converge with 5 component carriers whereas the real-valued algorithms require 7. This shows that the real-valued algorithms have a significant probability to get stuck in local minima in the modeled scenario. In Figure 4(a), the number of cell reboots per added BS is plotted. This plot is in good accordance with the plots in 3. The CDFs obtained using synchronization threshold $H_{\text{synch}} = -7$ dB are shown in Figure 4(b). From the figure it is visible that binary-pricing algorithms perform better than real-valued pricing ones, especially for users in the low C/I region.

In this case, the reporting load of the UEs was low, and the BSs have little information to base their decisions on. Thus, when the interference couplings are only statistically related to the typical user's C/I situation, using binary-valued pricing with plateau moves to attempt global optimization outperforms local optimization based on real-valued interference.

6. Graph Coloring for Automated Physical Cell ID Assignment

6.1. Performance of Distributed Algorithms. Performance is evaluated in the picocellular indoor office environment discussed in Section 4. The target is to find a number of PCIs that is sufficient to allow newly entering cells to configure their PCI in a confusion-and conflict-free manner, with a low level of cell-reboots required. The primary performance metrics are the same as in the previous section, except that instead of conflict freeness, confusion freeness is the target. Also, the C/I distribution is not considered. It is meaningless, as HO confusion is by definition a binary effect. The four distributed local search algorithms discussed in Section 3 are evaluated. In addition, the complete ABT and AWC constraint satisfaction algorithms are used.

For confusion freeness, we need to determine the prices used in the graph of two-hop neighbors. With binary pricing, the only issue is whether or not there is a confusion, that is, the binary and constraint satisfaction algorithms run directly on the two-hop neighbor graph. With real-valued pricing, a real interference distance is calculated for two-hop neighbors as the sum of the dB-scale real-valued interference price of both hops.

Additionally, the confusion couplings and their prices are assumed to be symmetrized. This can be understood as a by-product of the negotiations required to collect confusion information at the decision making node. For symmetrization purposes, or for the operation of the algorithm itself, a signaling channel between the neighboring cells is needed. In LTE systems, the X2 interface provides a natural candidate for this.

After the BS is switched on, it first scans for the synchronization channels of its neighbors, performing so-called

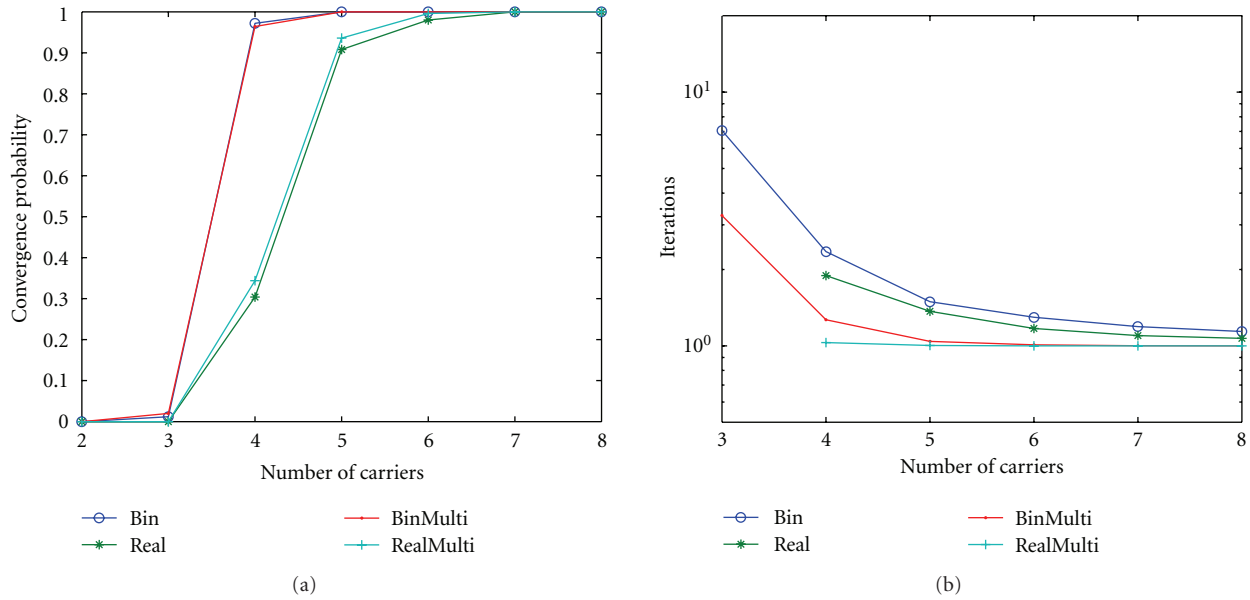

Figure 3: Convergence properties versus number of PCIs for different autonomous Primary Component Carrier Selection algorithms. (a) Convergence probability. (b) Number of iterations for convergence (only converged drops considered).

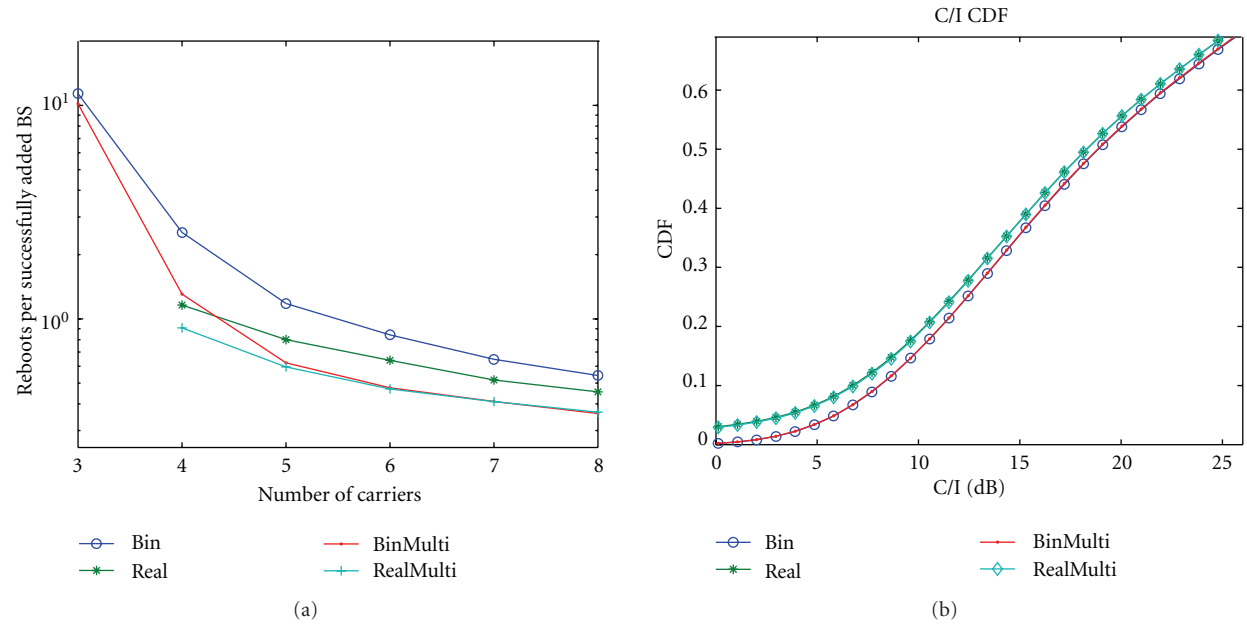

Figure 4: Performance of Primary Component Carrier Selection algorithms. (a) Number of cell reboots versus number of PCIs (only converged drops considered). (b) C/I CDFs for five component carriers.

over-the-air measurements. It finds neighbors with a C/I above H_{synch}. It selects a PCI randomly from the space of PCIs available, avoiding the ones that are used by the neighbors it is able to synchronize to. Note that this does not guarantee conflict freeness with first-hop neighbors, as the set of neighbors a BS can reliably synchronize to is smaller than the set of neighbors. After selecting an initial color, the BS starts serving UEs, collects HO candidate information from them, and performs an Automated Neighbor Relation (ANR) function to establish a connection with the neighboring BSs. If the BS selected a PCI causing a conflict with a neighbor, it is assumed that the neighbors of the conflicting pair are able to identify the conflict by ANR.

With an initial PCI chosen and neighbor relations established, conflict and confusion resolution is performed; a routine of a networking algorithm is executed at each node. If a confusion-free PCI configuration is identified, the results are stored, and a new BS is switched on. The algorithm runs until it converges, or until 1000 iterations have been done.

Related to conflict freeness, conclusions may be drawn from the analysis in the previous section. Thus, for example, a conflict-free assignment of PCI modulo 6, should be possible with the algorithms considered. This may be relevant for DL pilot orthogonalization, as discussed in Section 2. Considering confusion freeness, convergence probabilities for different number of PCIs are shown in Figure 5(a). As the maximum number of two-hop neighbors observed in the network is 26, it can be seen that it is possible to find a confusion-free PCI configuration with a significantly smaller number of PCIs. All algorithms except the random-try real-pricing algorithm converge within 1000 iterations for $N_{\text{PCI}} = 15$. The complete ABT algorithm performs worse than the best local search algorithms. The reason for this is that ABT relies on exhaustive search with a fixed ordering of the agents to find the solution, which is not possible with the limited number of iterations considered here, when the number of colors is small. Recall that the largest number of states in the system is an astronomical N_{PCI}^{96}.

AWC performs much better than ABT, and is clearly the best algorithm when it comes to convergence. AWC avoids exhaustive search by dynamic updating of priorities and minimum conflict heuristics. The binary-pricing algorithms are able to find a converged PCI for $N_{\text{PCI}} = 12$, as opposed to the multiple-try real-price algorithm $N_{\text{PCI}} = 15$, indicating a gain of ∼25%. From this, it is evident that when the target is to achieve confusion freeness, the property of the real-pricing algorithms to get stuck in a local optimum leads to nondesirable results.

In Figure 5(b), the average number of iterations for convergence is reported, where converged drops only are considered. It is notable that ABT and AWC do not fall down as rapidly as the local search algorithms with high N_{PCI}. The reason for this is the use of global IDs to prioritize in conflict situation, which forces the nodes to negotiate for a longer time to identify the node that should solve the conflict.

The number of cell reboots for converged drops are plotted in Figure 6. The multiple-try algorithms require a clearly lower number of reboots than the algorithms selecting a new PCI candidate randomly. This is natural, as the multiple-try algorithms always find a confusion-free PCI if available, and accordingly converge more rapidly. When comparing the number of reboots (for the converged drops) for the real- and binary-pricing algorithms, it can be seen that when the random-try real-pricing algorithm is able to find the global optimum, it does it with less reboots than the random-try binary algorithm. With the multiple-try algorithms, there is no difference in the number of reboots between the binary and real-pricing algorithms in the cases that all drops converge. Comparing to ABT and AWC shows that just as in the case of the number of iterations, these require more reboots than the best local search algorithms.

It should be noted that the fact that the number of reboots does not asymptotically vanish is due to the random initial selection of PCI for the switched on cell. There is always a nonzero probability that this initial selection is conflicting/confused with a neighbor/two-hop neighbor. However, the results in Figure 6 tell us that it is counterproductive to reserve a part of the PCI space for switching-on cells, as suggested in [8]. If that is done, there is always at least one cell reboot per added BS. Here, we see that with a sufficiently large space of PCIs, one get to significantly smaller number of reboots.

As a summary, these results point to a tradeoff when selecting a distributed confusion resolution algorithm. If one is extremely greedy related to the number of PCI's, one has to rely on the complete AWC algorithm, which requires an additional protocol to establish communication between cells that are not two-hop neighbors. If one is moderately greedy, one may do with a multiple-try local search algorithm based on binary confusion pricing.

6.2. Analysis of PCIs Required for Convergence. Here, we try to understand the relationships of the distributions of the number of neighbors and the number of two-hop neighbors in Figure 2, and how they affect the number of PCIs required for confusion freeness. These PDFs are crucial statistics for the behavior of different algorithms, and they are scenario-specific. In the scenario investigated here, for conflict freeness, we need at least 5 PCIs. To realize a confusion-free PCI configuration, we make a trivial observation from the PDF distributions that this can be achieved if we have 27 PCIs. With this number, which is close to the 30 different modulo 30 uplink sequence groups defined in LTE, we can easily configure the network in a distributed manner even in the worst scenario. However, reducing the number of required PCIs (or PCI groups) will allow certain flexibility on system design, as argued in Section 2. A more aggressive and yet straightforward observation is to take the mean value of the distribution as an estimation of the required PCIs. In the PDF of two-hop neighbors, we have the mean value around 10, while 11–15 PCIs are needed for convergence in our numerical results. By definition, two-hop neighbors are neighbors of the neighbors, implying that the distributions in Figure 2 are correlated. A node with more one-hop neighbors is likely to have more two-hop neighbors. We utilize this property for further discussion on the required number of PCIs.

For cells with different numbers of one-hop neighbors, it is more geometrically uniform if these cells are evenly distributed in the considered scenario. This means that for any cluster of cells, their statistics would be close to the statistics of the whole scenario. If this is the case, one can use first-order analysis to estimate the required number of PCIs. With uniform geometry, we can assume that each neighbor will on average introduce $\mathcal{X}$ new two-hop neighbors

$$\mathcal{X} = \overline{N_{1h}} \cdot p, \tag{5}$$

where $\overline{N_{1h}}$ is the mean number of one-hop neighbors and p is the probability of being a new two-hop neighbor. Using first-order moments of the distribution of one- and two-hop neighbors, one can train p by solving

$$\overline{N_{2h}} = \left(\overline{N_{1h}} - 1\right) + \left(\overline{N_{1h}} - 1\right) \cdot \mathcal{X}, \tag{6}$$

where $\overline{N_{2h}}$ is the mean number of two-hop neighbors. In (6), the first $\overline{N_{1h}} - 1$ comes from the fact that your first

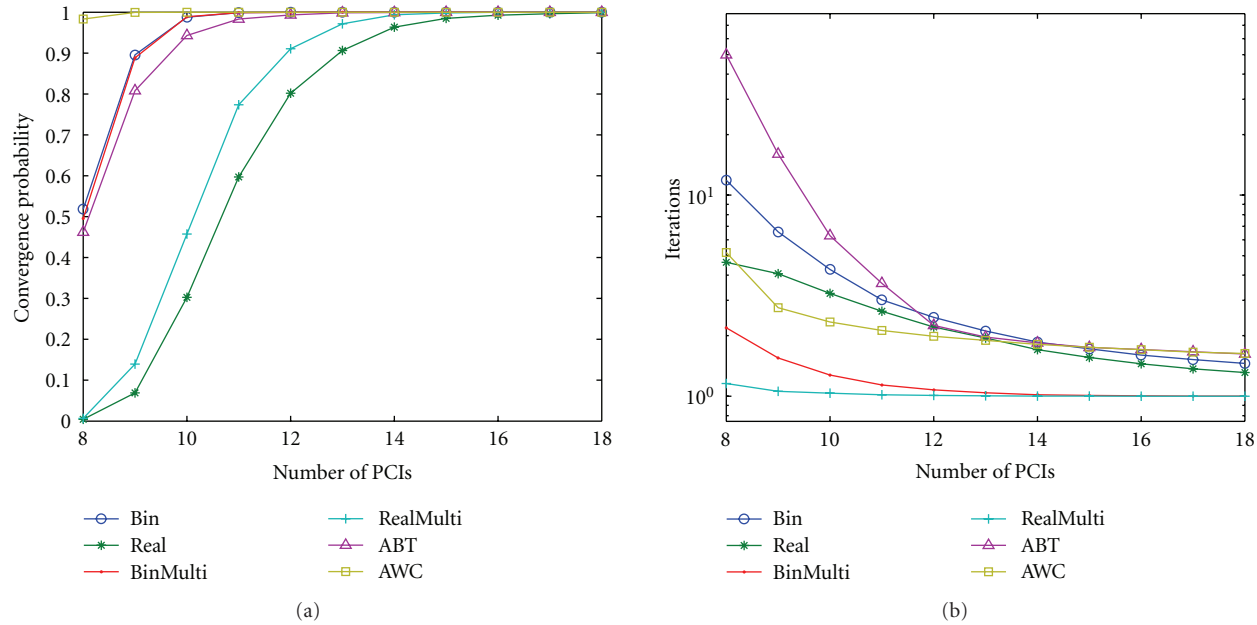

Figure 5: Convergence properties versus number of PCIs for different autonomous PCI configuration algorithms. (a) Convergence probability. (b) Average number of iterations (only converged drops considered).

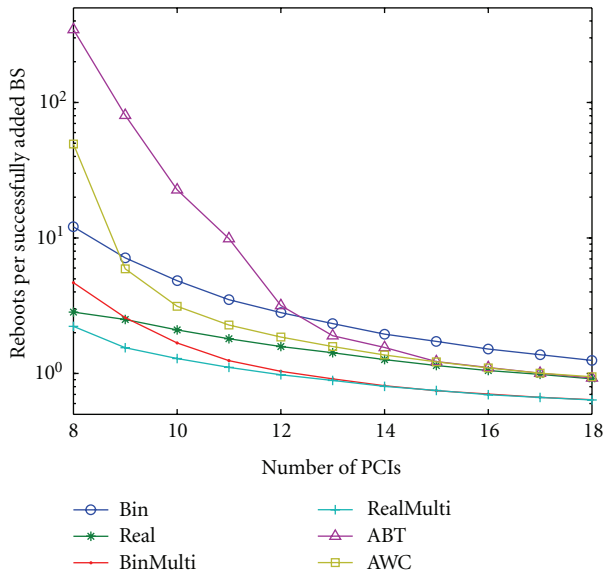

Figure 6: The number of cell reboots versus number of PCIs for different autonomous PCI configuration algorithms (only converged drops considered).

neighbor's neighbors are all new two-hop neighbors except for yourself. The second $\overline{N_{1h}} - 1$ considers the remaining one-hop neighbors.

After training p, the required PCIs can be estimated by calculating the additionally needed PCIs, out of those for avoiding conflict. We thus have the number of required PCIs as

$$N_{\text{PCI}} = N_{\max,1h} + (N_{\max,2h} - N_{\max,1h}) \cdot p, \quad (7)$$

where $N_{\max,1h}$ and $N_{\max,2h}$ are the maximum number of one- and two-hop neighbors, respectively.

On the other hand, if the statistics of clusters of cells is not similar with each other, cells with similar number of neighbors are geometrically closer to each other. In terms of PDF of two-hop neighbors, the higher end represents the statistics of a cluster with high number of one-hop neighbors. Thus, the tail at the higher end is weaker, as is the case in two-hop neighbor distribution of Figure 2. Besides, if we follow the same method as for uniform geometric case, one would see that p is higher than 1, indicating that the mean value of one-hop neighbors is not sufficient to support the mean value of two-hop neighbors. To accommodate this difference, we use similar procedure as uniform geometric case but use $N_{\max,1h}$ instead of $\overline{N_{1h}}$ in training p. This will enhance the fact that the higher end of two-hop PDF corresponds to cell clusters where each cell has higher number of neighbors. This results in $p = 0.3$ and therefore, the required number of PCIs is according to (7) is 12.

With this mechanism, it may be possible to develop a SON algorithm to determine the minimum number of PCIs required for confusion freeness just based on collecting statistics of the number of neighbors from the base stations.

7. Conclusion

We evaluated the use of distributed graph coloring algorithms for two self-configuration problems, pertinent to LTE-A; Primary Component Carrier Selection and Physical Cell ID Assignment.

Simple distributed algorithms based on local search were used for the self-configuration tasks. Real-valued coloring algorithms are based on a real interference price. Distributed

versions of such algorithms converge rapidly to a local optimum, where they are stuck. Discretizing the interference prices to binary conflicts, simple local search algorithms are able to move around on plateaus where the number of conflicts is constant, and then potentially find a globally optimum, conflict-free state.

For evaluating the distributed algorithms, an office Manhattan scenario was used. It was observed that despite the a priori high connectivity of the BSs, Autonomous Primary Component Carrier Selection works with a number of component carrier that is roughly half of the maximum number of neighbors, and Autonomous PCI Assignment works with a number of PCIs that is roughly half of the maximum number of two-hop neighbors. This is much less than the usually considered practical lower limit for distributed graph coloring, which is slightly larger than the order of the maximum number of neighbors [12].

Related to component carrier selection, the results lead to a conclusion that dividing a 100 MHz LTE-A system bandwidth to five component carriers of 20 MHz is a viable strategy, even if autonomous selection should be done.

For Physical Cell ID assignments, the results show that the space of PCIs can be significantly reduced without jeopardizing confusion freeness. For example, if a SON protocol for guaranteeing confusion-free PCI assignment is designed, it may well be designed to operate on a PCI modulo 30 basis, so that the reuse of UL reference signal sequence groups is automatically distributed to far-away cells. The results show that there is no gain from reporting other than binary confusion prices. Moreover, when N_{PCI} is large enough (larger than 14 in this example), it is counterproductive to have a space of temporary PCIs. Just selecting from the space of all PCIs randomly, the expected number of cell reboots is well below 1, which is the minimum when a temporary PCI is used. The price for this is that there is a small possibility that a cell which is not newly switched on has to change its PCI. Local search algorithms were contrasted to complete algorithms, that are distributed realizations of a search tree. It was observed that the best complete algorithm outperforms local search algorithms, when the number of PCIs is very small.

Acknowledgments

The work of F. Ahmed has been supported by the Academy of Finland (grant number 133652). The work of M. Peltomäki, J.-M. Koljonen and C.-H. Yu has been supported by Nokia.

References

[1] F. Lehser, Ed., "Next generation mobile networks— recommendation on SON and O&M requirements," Tech. Rep., NGMN Alliance, December 2008.

[2] M. Döttling and I. Viering, "Challenges in mobile network operation: towards self-optimizing networks," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '09)*, pp. 3609–3612, April 2009.

[3] ITU-R, "Guidelines for evaluation of radio interface technologies for IMT-Advanced," Tech. Rep. M 2135, 2008.

[4] 3GPP, "Feasibility study for further advancements of E-UTRA (LTE-Advanced)," Tech. Rep. TR 36.912, 2009.

[5] 3GPP, "Primary component carrier selection, monitoring, and recovery," Tech. Rep. R1-091371, 2009.

[6] L. G. U. Garcia, K. I. Pedersen, and P. E. Mogensen, "Autonomous component carrier selection: interference management in local area environments for LTE-advanced," *IEEE Communications Magazine*, vol. 47, no. 9, pp. 110–116, 2009.

[7] A. Eisenblätter, M. Grötschel, and A. M. Koster, "Frequency planning and ramifications of coloring," *Discussiones Mathematicae Graph Theory*, vol. 22, no. 1, pp. 51–88, 2002.

[8] Nokia Siemens Networks and Nokia, "SON use case: cell Phy ID automated configuration," Tech. Rep. R3-080376, 3GPP, 2008.

[9] T. Bandh, G. Carle, and H. Sanneck, "Graph coloring based physical-cell-ID assignment for LTE networks," in *Proceedings of the ACM International Wireless Communications and Mobile Computing Conference (IWCMC '09)*, pp. 116–120, June 2009.

[10] P. Kyösti et al., "Winner II channel models," Tech. Rep. D1.1.2 V1.2, 2007, http://www.ist-winner.org.

[11] 3GPP, "Evolved universal terrestrial radio access; physical channels and modulation (release 8)," Tech. Rep. TS 36.211 v8.6.0, 2009.

[12] F. Kuhn and R. Wattenhofer, "On the complexity of distributed graph coloring," in *Proceedings of the 25th Annual ACM Symposium on Principles of Distributed Computing (PODC '06)*, pp. 7–15, July 2006.

[13] M. Yokoo and K. Hirayama, "Algorithms for distributed constraint s: a review," *Autonomous Agents and Multi-Agent Systems*, vol. 3, no. 2, pp. 185–207, 2000.

[14] W. Zhang, G. Wang, Z. Xing, and L. Wittenburg, "Distributed stochastic search and distributed breakout: properties, comparison and applications to constraint optimization problems in sensor networks," *Artificial Intelligence*, vol. 161, no. 1-2, pp. 55–87, 2005.

[15] P. Galinier and A. Hertz, "A survey of local search methods for graph coloring," *Computers and Operations Research*, vol. 33, no. 9, pp. 2547–2562, 2006.

[16] J. O. Neel and J. H. Reed, "Performance of distributed dynamic frequency selection schemes for interference reducing networks," in *Proceedings of the IEEE Military Communications Conference (MILCOM '06)*, October 2006.

[17] B. Babadi and V. Tarokh, "A distributed asynchronous algorithm for spectrum sharing in wireless ad hoc networks," in *Proceedings of the 42nd Annual Conference on Information Sciences and Systems (CISS '08)*, pp. 831–835, March 2008.

[18] M. Kardar, *Statistical Physics of Fields*, Cambridge University Press, Cambridge, Mass, USA, 2007.

[19] M. Alava, J. Ardelius, E. Aurell et al., "Circumspect descent prevails in solving random constraint satisfaction problems," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 105, no. 40, pp. 15253–15257, 2008.

[20] M. A. Salido and F. Barber, "Distributed CSPs by graph partitioning," *Applied Mathematics and Computation*, vol. 183, no. 1, pp. 491–498, 2006.