Self-Configuration of Scrambling Codes for WCDMA Small Cell Networks

Alessandro Checco*, Rouzbeh Razavi†, Douglas J. Leith* and Holger Claussen†
*Hamilton Institute, NUI Maynooth. †Bell Laboratories, Alcatel-Lucent Ireland

***Abstract*—This paper introduces the problem of Primary Scrambling Code (PSC) selection in small cell networks and proposes a novel optimisation technique. Small cells introduce challenges not present in conventional macrocell scrambling code allocation, including the need for dynamic allocation, scalable distributed allocation algorithms, and support for unplanned and organic deployments. To the best of our knowledge this is the first study addressing the issue of distributed scrambling code selection for small cell networks. We propose a decentralized learning algorithm which does not require any collaboration between the neighbouring base-stations and which finds a feasible allocation whenever one exists. The performance of the algorithm is compared against two variations of a greedy algorithm which is the current 3GPP recommendation and is shown to offer significant performance benefits.**

***Index Terms*—Small Cell Networks, Femtocells, Primary Scrambling Code, Self-Configuration, Self-Optimisation**

I. Introduction

With the rapidly growing demand for mobile broadband services and the emergence of new high capacity mobile devices, mobile operators are struggling to meet the resulting capacity demand while keeping costs at an economic level. According to a recent Cisco report [1] mobile tablets will generate as much traffic in 2015 as the entire global mobile network in 2010. Small cells are now envisioned as one of the most promising solutions by the industry.

Reducing cell size is one of the simplest yet most effective solutions for capacity improvement e. g. in [2] the spectral gain efficiency of wireless systems from 1950-2000 is analysed and shows that shrinking of cell size has resulted in a factor of 2700 spectral efficiency gain. Self-configuration and self-optimisation are considered as a key enabler for successful deployment of small cells. This is especially true considering most deployments of small cells (e. g. femtocells) are expected to be ad-hoc and unplanned. The 3GPP standard [3] classifies the requirements for self-configuration of small cells into 10 categories where detailed fulfilment of these requirements is left to vendor specific solutions. The optimal selection of Primary Scrambling Codes (PSCs) is considered in this list and is additionally ranked as one of the top 5 most important parameters [3].

The PSC used by a base-station in UMTS and HSPA acts as an identifier and it is important that neighbouring base stations employ different scrambling codes in order to correctly manage handovers and cell association. The 3GPP requirements for macrocells therefore specify that direct neighbours and second order neighbours use different PSCs. In fact, detection of the PSC is an essential part of the target cell search procedure. In practice, a total of 512 available PSCs are divided into 64 groups each with 8 codes. The target cell procedure consists of two main stages where the User Equipment (UE) firstly determines the scrambling code group to which the target cell belongs and then in the second stage, it determines the PSC. Compared to the initial cell search, the target cell search is considerably simplified since the search space is restricted to the Neighbour Cell List (NCL).

In legacy macrocells, the PSC is chosen from a total of 512 available codes and, with such a large number of available codes, scrambling code allocation is a fairly straightforward task [4]. This task is mostly carried out manually or through centralised algorithms using cluster reuse-based techniques [5] or centralised graph colouring schemes [6]. While the NCL can be constructed manually with potential enhancement through drive tests to include missing neighbours by constructing the cell overlapping matrix for legacy macrocell networks [7], this would not be applicable to small cells considering their unplanned deployment. Therefore a number of PSCs are reserved specifically for small cells. Because small cells can be deployed within the coverage area of any macrocell, all macrocells need to add this set to their existing NCL. To avoid excessively long NCLs, the number of reserved PSCs is kept very small (3 or 4 in current implementations). A too long NCL leads to slower neighbour detection, measurement and cell acquisition [8].

For small cells, scrambling code allocation is much more challenging for four main reasons:

1) Small cells are not only constrained to choose from amongst only very few reserved scrambling codes but also are typically more densely deployed: it is not even guaranteed a non-interfering allocation exist;
2) Dynamic allocation is required due to the unplanned, organic nature of deployments;
3) Allocation algorithms are constrained to use little or no message passing between base stations in order to ensure scalability to large network sizes;
4) Since the reserved scrambling codes for small cells is a unique set that is added to all macrocells NCL, intelligent code planning techniques such as choosing the codes from the same code group or choosing identical codes from different groups becomes more difficult.

The main contributions of this paper are: (a) The formal

This material is based upon works supported by Science Foundation Ireland under Grant No. 07/IN.1/I901.

definition of the dynamic scrambling code allocation problem for small cells; (b) The design of a novel decentralised (with no message passing) algorithm for dynamic scrambling code allocation; (c) Analysis and performance evaluation of this algorithm using numerical simulations. The results show a dramatic improvement (up to 150% reduction in code confusion when a feasible solution exists) with respect to the 3GPP recommended scheme.

The rest of the paper is organised as follows. Section II formulate the problem of scrambling code allocation in general and introduces the notations used. Section III introduces the scrambling code selection algorithms including the proposed scheme of this paper. Subsequently, Section IV describes how the confusion graph can be constructed in practice. The simulation results are presented and discussed in Section V and finally Section VI concludes the paper.

II. Problem Statement

We introduce the following notation:

- Let $\mathcal{B}$ denote the set of small base-stations, with $|\mathcal{B}| = N$.
- Let $\mathcal{S}$ denote the set of PSCs reserved for small cells, with $|\mathcal{S}| = M$.
- Let $s_i \in \mathcal{S}$ denote the scrambling code associated to base-station $i \in \mathcal{B}$, and $s \in \mathcal{S}^N$ a global scrambling code allocation.
- Let $r_i(p)$ the received signal code power (RSCP) from the CPICH pilot of base-station $i \in \mathcal{B}$ measured at location $p \in \mathbb{R}^2$, and $r_{i,j} := r_i(p_j)$, where p_j is the location of device j (which might be a UE or another base-station).

Let $m_{i,j}$ denote the maximum ratio of received pilot power of base-station i to that of base-station j in the coverage area C_j of base-station j, i. e. $m_{i,j} := \max_{p \in C_j} \frac{r_i(p)}{r_j(p)}$. The *neighbor set* for base-station i is defined as $n_i = \{j : j \in \mathcal{B} \colon w_{j,i} > 0\}$, where

$$w_{j,i} = \begin{cases} m_{j,i} & \text{if } m_{j,i} > T \\ 0 & \text{otherwise,} \end{cases}$$

and T is an appropriate threshold value. We define the network *confusion graph* be the weighted directed graph $G := (\mathcal{B}, E, w)$ with vertices $\mathcal{B}$ and edges $E := \{(i,j) : j \in n_i\}$. That is, the graph with network base-stations as vertices and edge between each vertex/base-station i and every member of its neighbour set n_i. Each edge between two neighbours (i,j) is assigned weight $w_{i,j}$. Graph G is fundamental because if every vertex is assigned a different PSC from all of its neighbours then code confusion cannot occur. We call such an allocation a *proper* code allocation. Conversely, if any neighbours in this graph choose the same PSC then code confusion may occur.

We define the *utility* of a code allocation as:

$$U(s) = \frac{\sum_{i=1}^{N} u_i}{N}, \quad u_i = \begin{cases} 1 & \text{if } s_i \neq s_j, \text{ for all } j \in n_i \\ 0 & \text{otherwise.} \end{cases}$$

The utility of a proper code allocation s is $U(s) = 1$, while if all base-stations have at least one neighbour with the same PSC then $U(s) = 0$. Given a confusion graph G and a set $\mathcal{S}$ of codes, the code allocation task is to find a code allocation s with maximal utility $U(s)$. We say that this code allocation task is *feasible* if at least one proper choice of PSCs exists (and so the maximal $U(s) = 1$). Whether a feasible allocation exists depends both on the graph G and the number M of codes available.

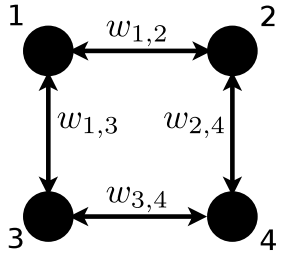


Figure 1. Simple example of a confusion graph G for a scenario consisting of 4 base-stations with symmetrical weights, and $w_{1,2} < w_{2,4}, w_{3,4} < w_{1,3}$.

As an example, Figure 1 shows a simple case of a confusion graph G for a network consisting of 4 base-stations where the weights associated to the edges are symmetrical, i. e. $w_{i,j} = w_{j,i}$.

III. Scrambling Code Selection

In this section we introduce three decentralized algorithms for tackling the code allocation task. All algorithms are run by the base-stations in an unsynchronised fashion, i. e. each base-station asynchronously updates its choice of PSC. We will discuss how the ordering of updates can affect the performance of the algorithms.

A. State of the Art: Single-step Greedy Algorithm (SGA)

The SGA algorithm is based on the current 3GPP recommendation [3] that is envisioned as a potential solution while other vendor specific methods are concurrently encouraged. Each base-station $i \in \mathcal{B}$ scans the set of available PSCs and determines the set $\mathcal{S}_{\text{allocated}}(b) = \{s_j, j \in n_i\}$ of PSCs used by its neighbouring base-stations. From this it determines the set $\mathcal{S}_{\text{free}}(i) = \mathcal{S} \setminus \mathcal{S}_{\text{allocated}}$ of unallocated PSCs. If $\mathcal{S}_{\text{free}}(i)$ is non-empty, then base-station i picks a PSC uniformly at random from $\mathcal{S}_{\text{free}}(i)$. Otherwise, base-station i selects the code that is used by a neighbour of minimal weight $s = \operatorname{argmin}_{s \in S} \max_{j \in n_i, s_i = s} w_{j,i}$.

While appealingly simple, it is important to note that this greedy algorithm is not guaranteed to find a proper code allocation even in simple scenarios and in general its performance may be poor. For example, consider the confusion graph shown in Figure 1, with weights chosen such that $w_{1,2} < w_{2,4}, w_{3,4} < w_{1,3}$ and suppose $M = 2$ PSCs are available for use by small cells, $\mathcal{S} = \{A, B\}$. It can be readily verified that a proper code allocation exists, namely assigning code A to vertices 1 and 4 and code B to vertices 2 and 3. However, in the SGA algorithm it may happen that vertex 1 chooses code A, then vertex 4 chooses code B and now vertices 2 and 3 are unable to choose a code that ensures a proper allocation (vertex 2 will choose code A, conflicting with vertex 1 and vertex 3 will choose code B, conflicting with vertex 4). Indeed in this case the utility $U(s) = 0$.

B. Iterative Greedy Algorithm (IGA)

A refinement is to execute the SGA repeatedly rather than just in a one-shot manner, thereby yielding the Iterative Greedy Algorithm (IGA). However, in general this suffers from similarly poor performance to the SGA e. g. it is easy to verify that in the previous example for SGA the allocation assigning code A to vertices 1 and 2 and code B to vertices 3 and 4 is a reachable equilibrium point of IGA for the confusion graph in Figure 1, i. e. there is a significant probability this algorithm converges to an allocation with $U(s) = 0$.

C. Communication-Free Learning (CFL) Algorithm

We now present the CFL algorithm which is related to the class of algorithms recently introduced in [9], [10]. Each base-station i maintains a vector p_i of length M. The s'th element $p_{i,s}$ gives the probability that base-station i chooses code s. Vector $p_{i,s}$ is updated according to Algorithm 1. In summary, when "satisfied" i. e. the current code choice is distinct from that of all neighbours n_i, base-station i keeps using this code. Otherwise, on "failure" it updates p_i to decrease the probability of selecting this code again and then randomly chooses a new PSC with probability given by vector p_i. The algorithm contains one parameter β which determines the algorithm's aversion to scrambling codes that lead to a failure: as β is made larger, failures are penalised more heavily and the memory or "stickiness" of the system decreases. The CFL algorithm is decentralised, meaning that it does not impose the requirement and overhead of communication between the base-stations. The implementation of the algorithm is especially straightforward making it suitable for small cells with limited memory and computing power.

Note that due to the stochastic nature of the CFL algorithm its convergence time is also stochastic. Unless otherwise stated, if the CFL algorithm takes longer than a given number of iterations to converge it is terminated early and the allocation given by scrambling code $\text{argmax}_j\, p_{u,j}$ is used for each station $b \in \mathcal{B}$.

Algorithm 1 Communication-Free Learning

Initialise $p_{i,s} = {}^1\!/_M \;\forall s \in \mathcal{S}$; select $\beta \in (0, 1]$
loop
  Select $s_i = s$ with probability $p_{i,s}$.
  if $s_i \cap \{s_j : j \in n_i\} = \emptyset$ "satisfied"

$$p_{i,s} = \begin{cases} 1 & \text{if} \quad s = s_i \\ 0 & \text{otherwise.} \end{cases}$$

  else

$$p_{i,s} = \begin{cases} (1-\beta)p_{i,s} & \text{if} \quad j = s_i \\ (1-\beta)p_{i,s} + {}^{\beta}\!/_{(M-1)} & \text{otherwise,} \end{cases}$$

  endif
end loop

IV. Confusion Graph Estimation

In order to decide the scrambling code, ideally each base-station i should have the knowledge of the RSCP levels $r_j(p), r_i(p), p \in C_i$ for all base-stations j. In practice, these quantities can be estimated based on UE reports. However, if the UE reports are to be taken into account, the base-station requires to allow long enough time to collect the UE measurements before deciding the scrambling code, which can lead to slow convergence of the network.

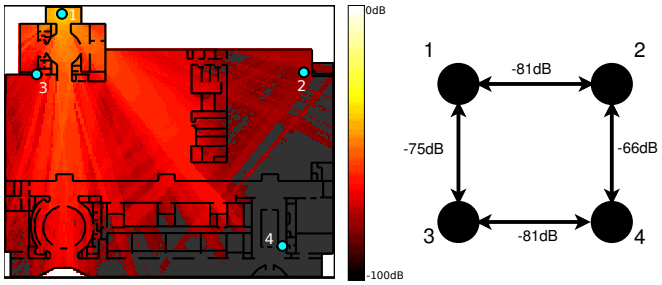


Figure 2. Example of small cell deployment within the Hynes convention centre in Boston: (a) Received pilot power for base-station 1, (b) Corresponding confusion graph G'.

A simpler way to construct the confusion graph is to rely on base-stations CPICH RSCP measurements of other base-stations. This corresponds, in our notation, to constructing the approximated confusion graph G' measuring $m_{i,j} := \frac{r_i(p_j)}{r_j(p_j)}$ for each base-station j. Similarly, the 3GPP recommendation [3] suggests to use base-stations measurements for scrambling code selection.

While the algorithms introduced in this paper can support both base-station based and UE-assisted graph construction, in order to make a fair comparison against the 3GPP scheme, the confusion graphs are constructed based on base-station measurements.

As an example, Figure 2-(a), illustrates the scenario where four base-stations are deployed within the conference hall at the Hynes convention centre in Boston. The heat-map shown in this graph refers to CPICH RSCP of base-station 1 (top-left) when the transmit pilot power is set to 10 dBm. The map is generated using the Wireless System Engineering (WiSE) [11] software which is a comprehensive 3D ray tracing based simulation package. The simulation results from WiSE have been validated by comparison against measurement data in a number of indoor and outdoor environments including the Hynes convention centre considered in this example. Figure 2-(b) shows the resulting approximated confusion graph G'.

V. Results - Numerical Simulations

Simulations were performed to evaluate and compare the performance of these three scrambling code allocation algorithms. Due to the complexity of the WiSE package and need for exhaustive global search for scrambling code allocation in many simulation scenarios, a simpler model was used. Varying number of base-stations are placed uniformly at random in a $100 \times 100\,\text{m}^2$ area. The update ordering of the base-stations is selected uniformly at random from the set of permutations of $\mathcal{B}$. The maximum transmit power is 100 mW and the pilot power is 10% of this value [12]. Radio propagation path loss used is the 3GPP model for small cells [13]. Simulation results are averaged over 100 independent runs.

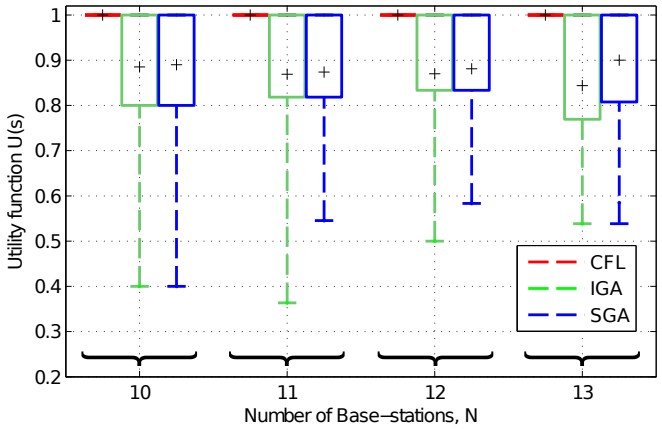


Figure 3. Comparison of SGA, IGA and CFL algorithms performance when a feasible code allocation exists.

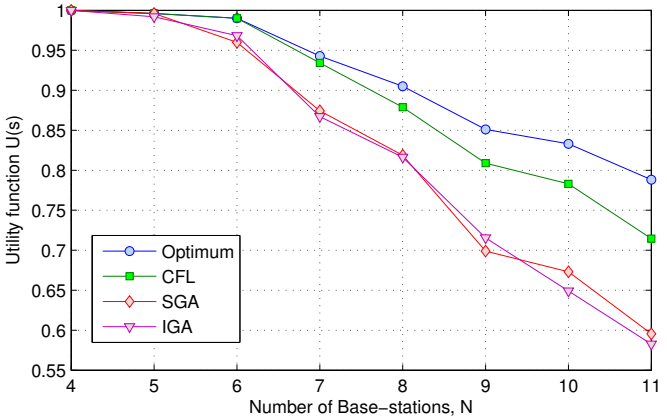


Figure 5. Mean utility function $U(s)$ versus number of base-stations for optimal scrambling code allocation and SGA, IGA and CFL algorithms when $M = 4$.

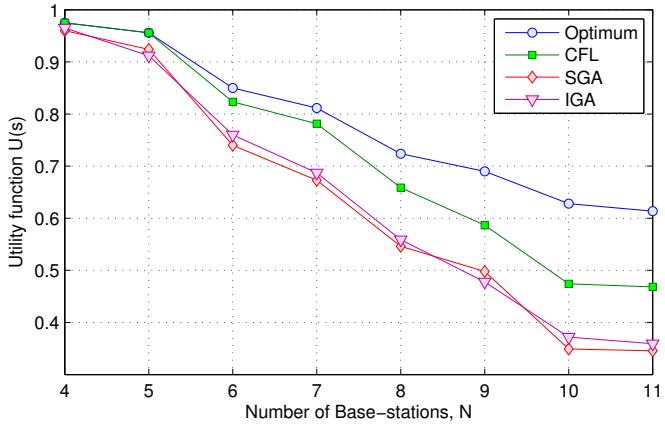


Figure 4. Mean utility function $U(s)$ versus number of base-stations for optimal scrambling code allocation and SGA, IGA and CFL algorithms when $M = 3$.

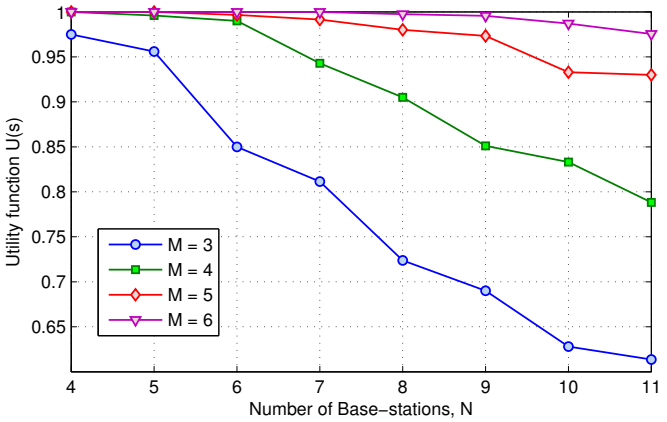


Figure 6. Mean utility function $U(s)$ corresponding to the optimal allocation for different number of base-station and varying number of scrambling codes, M.

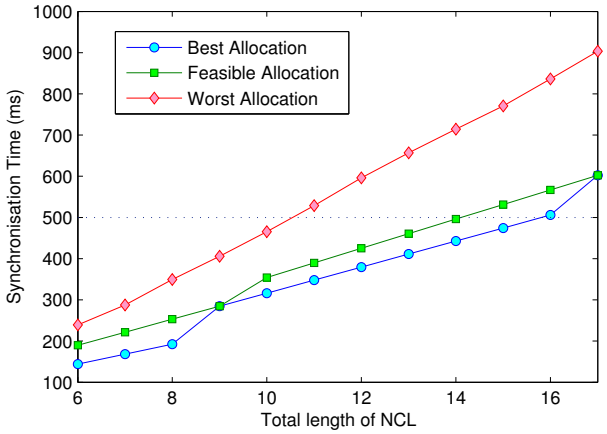
A. *Comparison of Scrambling Code Selection Algorithms*

Figure 3 shows the boxplot of the utility function, $U(s)$, for all three algorithms and for scenarios where a feasible scrambling code allocation exist (choosing the number of available codes M equal to the chromatic number of the confusion graph). In this figure, markers indicate the mean value of utility function, the bottom and the top of the boxes indicate the 25th and 75th percentile of the utility function samples. The figure additionally indicates the minimum values underneath the bottom of the boxes. Without exception, the CFL always finds a proper solution ($U(s) = 1$). In contrast, both IGA and SGA achieve mean utility of around 0.85 and also have relatively large degree of dispersion around this mean. The improvement of the utility function is up to 150%. Figure 4 shows the utility function versus the number of base-stations. The results are illustrated for the scenario when $M = 3$ scrambling codes are reserved for small cells (and so a proper code allocation may not exist). While the overall trend is expectedly descending, the CFL is shown to outperform the other two greedy algorithms by up to 50%. Additionally, the optimal allocation has been calculated with exhaustive search and compared against the CFL where the CFL is shown to perform relatively close to the optimal especially when considering practical densities. It should be note that the CFL performance is considered to be acceptable given that the code allocation problem is NP-hard and that CFL is a fully distributed algorithm which relies on individual base-station decisions without any message passing (i. e. no centralised knowledge of the network topology is available). Similarly, Figure 5 shows the results for the case when the number of scrambling codes is increased to 4. Compared to Figure 4, the results show significant improvement in terms of the average utility function as a result of the added scrambling code. Again, CFL is shown to be superior to both SGA and IGA algorithms and the performance is even closer to the global optimum.

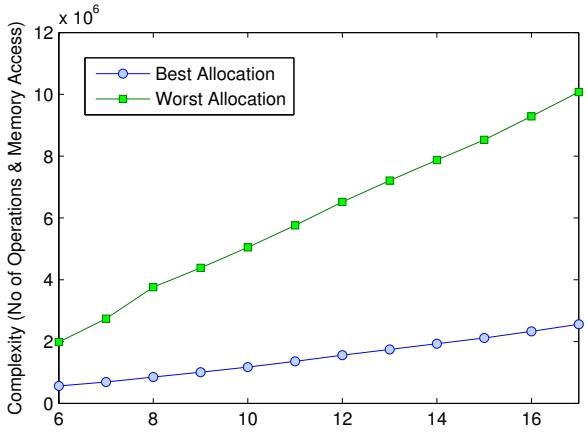
B. *Impact of NCL size*

To provide a better understanding of the effect of the number of available scrambling codes, Figure 6 shows the optimal utility function averaged over 100 simulation runs for increasing number of base-stations and varying number of available scrambling codes. As the results shows, increasing the number of scrambling codes has a significant effect on the

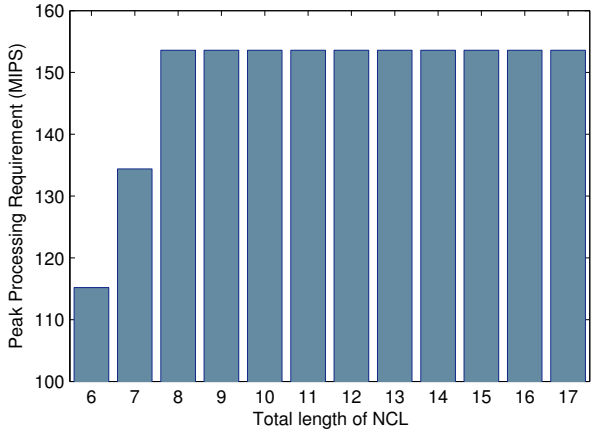
supported density of small cell deployments. The downside would evidently be the increased size of the NCLs. Using



(a)



(b)



(c)

Figure 7. (a) Synchronisation time (b) Associated complexity (total number of operations and memory access) and (c) peak processing requirement as a function of NCL size.

the model introduced in [14], Figure 7-(a) shows the overall synchronization time required for target cell measurements as function of the NCL size. In fact, the time required for synchronization depends on how scrambling codes are allocated between the neighbouring cells. In general, there is a trade-off between the synchronization time and the complexity (in terms of the number of operations and memory access) when scrambling codes are planned. While the synchronization time strongly depends on the number of code groups within a given NCL, the complexity rises with the number of individual codes [14]. The reason for this is because the final code detection is more robust than the code group detection stage albeit at the cost of increased complexity.

Moreover, poor code group detection degrades the overall synchronization process substantially and results in lengthy code acquisition. As an example, for an NCL of size 12, if 6 identical scrambling codes are selected from 2 groups, the required synchronization time per cell is 31.9 ms with the associated complexity of 2.59e5 operations and memory access. The corresponding values are 40.2 ms and 1.43e5 operations if the 2 codes are selected from 6 groups.

Since the codes that are decided for small cells ought to be fixed, it may not be possible to choose them from the same group as the macrocells. However, it is still possible to select the codes dedicated to small cells from the same group. In Figure 7-(a), considering the synchronization time as the performance metric, the worst case allocation refers to when all codes are selected from different groups and the best allocation corresponds to choosing codes from minimum possible number of code groups.

Given that each code group consists of 8 codes, there is a sharp jump in the synchronization time for the best allocation when the NCL size is increased from 8 to 9 (or from 16 to 17) which imply the need for a new code group. Furthermore, the feasible allocation refers to the case when the code group of which small cell codes are selected is different to that of the macrocells whilst the number of code groups is kept at minimum.

Given that the measuring period is 500 ms [14], Figure 7-(a) confirms that the size of the NCL should not exceeds 14 cells for heterogeneous networks. Therefore, the number of small cells reserved codes depends on the number of codes allocated to macrocells. While with an ideal hexagonal shape macrocells' layout, merely 6 codes are sufficient to represent the neighboring cells, in reality, a NCL size of 10 for macrocells is still not too conservative. This is especially the case considering dense deployment of the macrocells in urban areas. The exact number of required codes, depends on the network topology including the base-stations' location, their antennas' orientations and their pilot power settings.

Figure 7-(b) shows the complexity of the target cell search procedure as a function of the number cells within a given NCL. In contrary to Figure 7-(a), the best allocation here refers to when identical codes are selected all from different groups and worst allocation represents the scenario when many individual codes (up to 8) are selected used. While it is desirable to minimize the number of operations and memory access to allow longer battery life of UEs, this will not be a prime performance concern as long as UEs can handle the computation complexity.

For this reason, the worst case peak processing requirement, in million instruction per second (MOPS), was calculated and

shown in Figure 7-(c). Since the peak processing complexity depends on the number of individual codes, it remains the same after the NCL size exceeds 8. Fortunately, considering the processing capabilities of mobile phones [15], the worst case peak processing requirement of 153 MIPS is still well bellow almost all mobile phones. The margin is over 92% when considering more recent phones.

Comparing the results shown in Figure 7, it is evident that for heterogeneous networks consisting of small cells, the best scrambling code allocation strategy is to choose the codes from a minimum possible number of code groups.

VI. Conclusions

This paper introduces the problem of scrambling code allocation for WCDMA small cell networks. The problem differs from code planning in the legacy macrocell networks due to the limited number of codes reserved for small cells, the need for dynamic adaptation and for scalable, distributed allocation algorithms. Additionally, a novel decentralised (with no message passing) algorithm for dynamic scrambling code allocation is proposed and its performance was evaluated against two variants of 3GPP recommended schemes. The results confirm significant performance improvement of the utility function (up to 150%) when using the proposed scheme. The proposed scheme is fully distributed and is of low computation complexity which makes it suitable for unplanned deployment of small cells base-stations.

Finally, the paper discuss the trade-offs involved in increasing the number of codes reserved for small cells. While there is a significant improvement in term of supported small cell deployment density when the number of reserved codes is increased to 5 or 6, results shows that the NCL size may not exceeds a total of 14 cells for a synchronisation time of 500 ms. Additionally, the paper confirms the importance of scrambling code planning for cells belonging to a given NCL. Since the synchronisation time is shown to be the prime limiting factor, the best code allocation strategy for heterogeneous networks is the one that results in minimal number of code groups.

References

[1] C. V. N. Index, "Global Mobile Data Traffic Forecast Update 2010–2015," 2011.

[2] W. Webb, *Wireless Communications: The Future*, Wiley, Ed., New York, 2007.

[3] 3GPP TR 25.967, "Home Node B (HNB) Radio Frequency (RF) requirements (FDD)," vol. Ver 10, no. version 10, 2011.

[4] A. T. H. Holma, *WCDMA for UMTS - radio access for third generation mobile communications*, J. Wiley, Ed., Chichester, 2000.

[5] C. R. Chang, J. Z. Wan, and M. F. Yee, "PN offset planning strategies for non-uniform CDMA networks," in *Vehicular Technology Conference, 1997 IEEE 47th*, vol. 3, pp. 1543–1547.

[6] Y. Y.H. Jung, "Scrambling code planning for 3GPP W-CDMA systems," in *Proceedings of IEEE vehicular technology conference*, Rhodes, Greece, Spring 2001, pp. 2431–2434.

[7] I. O. D. Soldani, "Self-optimizing neighbour cell list for UTRA FDD networks using detected set reporting," *IEEE VTC2007-Spring2007*, pp. 694–698, 2007.

[8] J. C. J. Guey, Y. Wang, "Improving the robustness of Target Cell Search in WCDMA Using Interference Cancellation," 2005.

[9] K. R. Duffy, C. Bordenave, and D. J. Leith, "Decentralized Constraint Satisfaction," *ArXiv e-prints*, mar 2011.

[10] D. Malone, P. Clifford, D. Reid, and D. J. Leith, "Experimental implementation of optimal WLAN channel selection without communication," in *New Frontiers in Dynamic Spectrum Access Networks, 2007. DySPAN 2007. 2nd IEEE International Symposium on*, 2007, pp. 316–319.

[11] S. Fortune, D. Gay, B. Kernighan, O. Landron, R. Valenzuela, and M. Wright, "WISE design of indoor wireless systems: practical computation and optimization," *Computational Science & Engineering, IEEE*, vol. 2, no. 1, pp. 58–68, 1995.

[12] 3GPP TS25.101, "UE Radio transmission and Reception (FDD)," 2004.

[13] 3GPP TR 36.814, "Evolved Universal Terrestrial Radio Access (E-UTRA); Further advancements for E-UTRA physical layer aspects," vol. Rel 9, 2010.

[14] S. Kourtis, "Code planning strategy for UMTS-FDD networks," in *Vehicular Technology Conference Proceedings, 2000. VTC 2000-Spring Tokyo. 2000 IEEE 51st*, vol. 2. IEEE, 2000, pp. 815–819.

[15] D. Aguilar, "A framework for evaluating the computational aspects of mobile phones," Ph.D. dissertation, University of South Florida, 2008.