Real-Time Scheduling
for Content Broadcasting in LTE

Francesco Malandrino
Politecnico di Torino
Torino, Italy
Email: malandrino@tlc.polito.it

Claudio Casetti
Politecnico di Torino
Torino, Italy
Email: casetti@tlc.polito.it

Carla-Fabiana Chiasserini
Politecnico di Torino
Torino, Italy
Email: chiasserini@tlc.polito.it

Siyuan Zhou
Politecnico di Torino
Torino, Italy
Email: siyuan.zhou@polito.it

Abstract—Broadcasting capabilities are one of the most promising features of upcoming LTE-Advanced networks. However, the task of scheduling broadcasting sessions is far from trivial, since it affects the available resources of several contiguous cells as well as the amount of resources that can be devoted to unicast traffic. In this paper, we present a compact, convenient model for broadcasting in LTE, as well as a set of efficient algorithms to define broadcasting areas and to actually perform content scheduling. We study the performance of our algorithms in a realistic scenario, deriving interesting insights on the possible trade-offs between effectiveness and computational efficiency.

I. INTRODUCTION

LTE and LTE-Advanced networks have been conceived and designed for the purpose of facing an ever-increasing demand for capacity. Indeed, smartphones and tablets are now full-fledged entertainment stations, capable of displaying high-quality multimedia content – and their owners seem to love that. The days when the expression "mobile multimedia" referred to playing hiccup-plagued cat videos from YouTube are long gone; users demand low-latency multiplayer gaming, real-time video uploading and, increasingly, high-definition streaming.

Streaming is an especially challenging use case, in that it requires both high speed and low latency. Too many users playing the same content can choke even a high-capacity network such as LTE. Even high-capacity networks such as LTE can have trouble supporting too many users playing the same stream. Several approaches have been proposed to tackle this challenge [1], [2]. In addition to general-purpose techniques such as heterogeneous networking, there is a proposed feature of LTE that targets exactly the issue of real-time streaming – *broadcasting*. The intuition behind it is simple: operators can decide to devote a part of their spectrum resources to broadcasting high-demand content. Users requesting the content will be served without further increasing the network load, just as it happens with DVB television. Small-scale experiments involving broadcasting to mobile devices through LTE have been successfully carried out [3], and mobile operators are planning to employ this technology for massively popular sport events such as the Super Bowl [3]. More specifically, operators have to decide:

- *whether* to broadcast a certain content at all;
- *when* to broadcast it, accounting for its current (and future) popularity;
- *where* to broadcast it, as popularity is typically location-specific.

Such decisions must be taken in a clever way – broadcasting a content that is not sufficiently popular implies wasting precious spectrum resources, which could be used to serve ordinary, i.e., unicast, traffic. Of equal importance, and perhaps of a more challenging nature, decisions must be swift.

Taking swift decisions concerning LTE broadcasting is difficult for several reasons. The most obvious is that the elements to account for – potential content, associated demand, unicast traffic – change rapidly over time. Furthermore, the decisions that have to be taken are complex: deciding to broadcast a content in a cell has far-reaching consequences in terms of interference on the neighboring ones. Finally, there are technology- and standard-related constraints to honor, e.g., concerning the maximum amount of resources that can be devoted to broadcasting.

The solutions that have appeared in the literature so far have aimed at solving the problem of where massively-popular content should be broadcasted [4], [5]. In addition to network configuration, significant attention has been devoted to scheduling and resource allocation [6]. Indeed, in LTE broadcasting, UEs can send feedback about their perceived quality of service, and such information can be leveraged to adjust the scheduling over time. Finally, the white paper in [7] describes an early implementation of LTE broadcasting, and its ability to improve the network capacity and performance.

In this paper, we chart the path for the broadcasting of non-massively-popular content on LTE networks. Our contribution is twofold. First, we present a model for broadcasting in LTE networks. Simple and compact as it is, our model can capture all the decisions that have to been taken, their consequences and implications, and the constraints they are subject to. After discussing the impracticality of solving such a model to the optimum, we make our second contribution: a family of scheduling algorithms that are:

- effective, in that their output is close to the optimum;
- efficient, in that such an output is computed in a short time;
- informed, in that they account for the consequence of scheduling decisions on unicast traffic, as well as for the existing constraints.

Finally, we assess the effectiveness of our algorithms in a large-scale, realistic scenario.

The remainder of this paper is organized as follows. We describe how broadcasting is implemented in LTE standards in Section II. We present our model in Section III, and our algorithms in Section IV. We study their effectiveness in Section V. Lastly, Section VI concludes the paper.

II. BROADCASTING IN LTE

3GPP has introduced MBMS (Multimedia Broadcast and Multicast Service) as a point-to-multipoint way to broadcast and multicast data to mobile users, in release R6 [8]. In LTE system, MBMS has evolved into enhanced MBMS, with the introduction of Single-Frequency Networks (SFNs), in release R9 [9]. All eNBs belonging to the same SFN transmit the same information (the same bits) on the same carrier frequencies (licensed bands), in a synchronized fashion. This prevents interference within the same SFN.

Each SFN can span multiple *contiguous* cells; the set of cells belonging to the same SFN is called *MBSFN area* (Multimedia Broadcast over SFN). The maximum number of allowed MBSFN is 256 per geographical region.

Time multiplexing is another important aspect. A first constraint is that at most 6 out of 10 subframes can be used for broadcasting. Furthermore, UEs cannot be expected to receive data from multiple MBSFN areas at the same time. However, UEs can belong to multiple areas; it follows that the schedules of overlapping MBSFN areas cannot overlap.

III. SYSTEM MODEL

Our model focuses on a single time frame, during which it is reasonable to assume that aspects such as time variations in content popularity and user mobility do not vary. Without loss of generality, we assume that in each broadcasting area, it is broadcast exactly one piece of content. This means that, if we need to broadcast more than one content in the same cells, multiple areas will be created.

A. Building blocks

1) Cells and areas: There are three main components of our system: cells, broadcast areas, and content.

Cells $c \in \mathcal{C}$ are standard LTE cells. We call $\mathcal{E} \subseteq \mathcal{C}^2$ the set that contains all pairs of neighboring cells; thus $(c_1, c_2) \in \mathcal{E}$ if c_1 and c_2 are neighbors. To simplify the notation, we write the set of neighbors of cell c as $\mathcal{N}^c = \{c' \in \mathcal{C} : (c, c') \in \mathcal{E}\}$. Notice that c is considered a neighbor of itself, i.e., $c \in \mathcal{N}^c$. Also, in each cell c there are a total of U^c users.

Areas $a \in \mathcal{A}$ are the broadcast areas that we create and correspond to the MBSNF areas in LTE. To comply with LTE limitations, it should be: $|\mathcal{A}| \leq 256$. Clearly, the size of any area cannot exceed the total number of cells in the region, i.e., $|a| \leq |\mathcal{C}|$.

2) Content and popularity: We denote by $m \in \mathcal{M}$ the content items we may decide to broadcast, e.g., live events. For each cell c and content item m, we know the popularity π_m^c, i.e., the number of users in cell c interested in content item m at the current time.

3) Resources: Spectrum resources correspond to LTE resource blocks (RBs), and represent the usage we are making of the LTE spectrum. For each content item m, we know the amount ρ_m^c of resources needed to broadcast item m in cell c. Such an amount depends on both the content (e.g., the video resolution) and the cell, e.g., propagation conditions experienced by its users.

The number of existing resources is R, out of which at most $r \leq R$ are available for broadcasting. This allows us to represent, e.g., the 6/10 limit discussed in Section II.

B. Decision variables

We have two main decisions to take: which cells belong to each area, and which content is broadcasted in each area. To this end, we define two decision variables:

- a binary variable $\alpha_a^c \in \{0, 1\}$, expressing whether cell $c \in \mathcal{C}$ belongs to area $a \in \mathcal{A}$;
- a discrete variable $\mu_a \in \mathcal{M}$, expressing which content is broadcasted in area $a \in \mathcal{A}$.

Furthermore, we define an auxiliary variable x_a, expressing the amount of resources we use for broadcasting within area $a \in \mathcal{A}$. As discussed next, we need this variable in order to account for technology and standard constraints.

C. Constraints

The first constraint we need to impose concerns the minimum amount of resources ρ_m^c. If cell c belongs to area a, i.e., $\alpha_a^c = 1$, then area a must use enough resources to properly stream its content μ_a to cell c:

$$x_a \geq \rho_{\mu_a}^c, \forall a \in \mathcal{A}, c \in \mathcal{C} : \alpha_a^c = 1. \qquad \text{(III.1)}$$

Next, we account for the total amount r of resources that can be devoted to broadcasting. For each cell $c \in \mathcal{C}$, the sum of the x-values of the cells it belongs to cannot exceed r:

$$\sum_{a \in \mathcal{A} : \, \alpha_a^c = 1} x_a \leq r, \forall c \in \mathcal{C}. \qquad \text{(III.2)}$$

Note that constraint (III.2) also poses a soft limit to the number of areas a cell can belong to.

Finally, we have to deal with interference. The most conservative approach would impose that if a resource is used by area a in cell c, then it should not be used by any other area overlapping either c or any cell neighboring c. A softer constraint is given by:

$$\sum_{a \in \bigcup_{c' \in \mathcal{N}^c} \{a \in \mathcal{A} : \, \alpha_a^{c'} = 1\}} x_a \leq r, \forall c \in \mathcal{C}. \qquad \text{(III.3)}$$

As complex as it looks, constraint (III.3) has a simple meaning: for each cell c, the areas to which c or any of its neighbors belong should have enough resources available so that a disjoint set can be scheduled. Recall that c is included in $\mathcal{N}^c$ by definition.

D. Performance metric and objective

Intuitively, our goal is to set the α- and μ-variables so as to maximize the system performance. However, the definition of "system performance" is rather vague, and deserves a deeper discussion.

Let us focus on a cell c, with U^c users within it. Also, let $\mathcal{A}^c = \{a \in \mathcal{A}: \alpha_a^c = 1\}$ be the set of areas c belongs to, and $\mathcal{M}^c = \{\mu_a, \forall a \in \mathcal{A}^c\}$ the set of content broadcasted therein.

We can identify three distinct groups of users:
- users that are served through broadcasting;
- users that would like broadcast-able content, but are not served by broadcasting;
- users that want to download unicast content.

For each group of users, we can compute a satisfaction metric.

Users that are served through broadcasting have satisfaction 1. The number of such users is given by $\sum_{m \in \mathcal{M}^c} \pi_m^c$. The remaining users are served through unicast. The pool of resources that can be assigned to them is given by the total amount R, minus the ones used by the area(s) that cell c belongs to, minus the ones interfered by neighboring cells: $R - \sum_{a \in \mathcal{A}: \sum_{c' \in \mathcal{N}^c} \alpha_a^{c'} = 1} x_a.$

With these resources, we have to serve the users that request content in $\mathcal{M} \setminus \mathcal{M}^c$, i.e., not transmitted through broadcast. The total amount of resources needed by these users is $\sum_{m \in \mathcal{M} \setminus \mathcal{M}^c} \pi_m^c \rho_m^c$. By denoting the number of unicast users and their resource request in cell c by π_u^c and ρ_u^c, respectively, the average number of satisfied users is given by:

$$\frac{R - \sum_{a \in \mathcal{A}: \sum_{c' \in \mathcal{N}^c} \alpha_a^{c'} = 1} x_a.}{\sum_{m \in \mathcal{M} \setminus \mathcal{M}^c} \pi_m^c \rho_m^c + \pi_u^c \rho_u^c}.$$

Combining the above expressions, we can define the following performance metric:

$$V^c = \sum_{m \in \mathcal{M}^c} \pi_m^c + \frac{R - \sum_{a \in \mathcal{A}: \sum_{c' \in \mathcal{N}^c} \alpha_a^{c'} = 1} x_a.}{\sum_{m \in \mathcal{M} \setminus \mathcal{M}^c} \pi_m^c \rho_m^c + \pi_u^c \rho_u^c}. \quad \text{(III.4)}$$

Equation (III.4) takes values between 0 and the number of users on our topology, and it represents the average total number of satisfied users.

IV. OUR APPROACH

As mentioned earlier, we have two main tasks to perform:
- assigning the cells to the areas;
- deciding the content to broadcast in each area.

Jointly tackling these tasks would require solving a MILP problem that, as discussed above, is intractable for realistic instances of the problem.

Therefore, we resort to a *divide-et-impera* approach [10], and decouple the two tasks. Specifically, we present a simple, efficient way to select the content μ_a to broadcast in an area a given the cells belonging to it, i.e., the α_a^c values. We leverage such an assignment technique and reduce our scheduling problem to assigning cells to the area, i.e., setting the α-values.

A. Selecting the content

Here, we assume we already know the cell-to-area assignment, i.e., the α_a^c-values for all areas $a \in \mathcal{A}$ and cells $c \in \mathcal{C}$. Our task is to determine the content $\mu_a \in \mathcal{M}$ to broadcast in each area.

We proceed in a straightforward way, as summarized in Algorithm 1. We begin by ranking areas by the number of users interested in broadcastable content that they include (line 1). Then, for each area starting from the biggest one, we identify the viable content, i.e., content that can be broadcasted to that area without violating constraint (III.3) (line 3). Finally (line 6), we select the viable content that maximizes the overall performance, as defined in (III.4).

Notice that it is possible (line 5) that the set of viable content is empty, i.e., no content can be broadcasted in the area. A typical reason for this is that all the 6/10 subframes available for broadcasting are occupied by other areas overlapping (or neighboring) with the current one. In this case, we simply proceed with the next area.

Algorithm 1 Assigning content to areas

Require: $\mathcal{C}, \mathcal{A}, \{\alpha_a^c\}$
1: `sort` $a \in \mathcal{A}$ by no. of users interested in broadcasting
2: **for all** $a \in \mathcal{A}$ **do**
3:     viable_content_set$\leftarrow \{m \in \mathcal{M}:$ (III.3) holds$\}$
4:     **if** viable_content_set$\equiv \emptyset$ **then**
5:         **continue**
6:     $\mu_a \leftarrow \arg\max_{\text{viable\_content\_set}} V^c$

In Algorithm 1 we follow a greedy approach, i.e., we never reconsider decisions once they are taken. This means that we have no formal optimality guarantee. However, starting from the areas with the highest number of users interested in broadcastable content, guarantees that any conflicts are solved in such a way that the largest number of users is satisfied.

Finally, we remark that, while solving the MILP formulation to the optimum in small-scale scenarios, we noticed that the selection of content μ_a has a smaller impact on the system performance than the cell-to-area assignment. It is thus preferable to employ a straightforward approach for selecting content, as we did, and a more sophisticated solution for the area formation.

B. Forming the areas

Different cells have, in general, different demand for different content. Intuitively, the most straightforward action one could take is forming as many areas as there are cells, with each area comprising one cell. Two factors concur in rendering such a straightforward solution undesirable and, in the general case, infeasible: the maximum number of areas that can be created, e.g., 256, and the inter-area interference.

The first aspect is clear: there is a hard limit on the number of areas we can form. Inter-area interference is a bit more complex. As mentioned in Section II, areas are implemented as single-frequency networks; therefore, there is no interference

between cells belonging to the same area. Neighboring areas, instead, are subject to interference; we model this through constraint (III.3) in Section III.

It follows that if we have two neighboring cells with similar (albeit not identical) content popularity values π_m^c, it is often better to put them in the same area (and serve only the content item that is popular in both cells) than having two separate areas whose schedules are tailored to each cell.

There is an essential tradeoff between two choices: small areas with high interference or bigger areas with less interference but broadcasting less popular content – we have to deal with when forming the broadcasting areas. There are two main approaches we can adopt to solve the problem, which we name *merge* and *grow*.

1) The merge *approach:* The intuition behind this approach comes directly from the above discussion. We start from an assignment where we create an area per cell (line 3). Then, we merge neighboring areas so as to maximize the (immediate) performance improvement (line 6). We stop when both the following conditions are met: first, the number of areas is below the maximum limit $\hat{A}$ (i.e., 256); second, there are no more pairs of areas that can be merged increasing the performance (line 8). More formally, we proceed as shown in Algorithm 2.

Algorithm 2 Merge approach

Require: $\mathcal{C}$
1: $\mathcal{A} \leftarrow \emptyset$
2: **for all** $c \in \mathcal{C}$ **do**
3:     $a \leftarrow \{c\}$
4:     $\mathcal{A} \leftarrow \mathcal{A} \cup \{a\}$
5: **while** True **do**
6:     $(a_1, a_2) \leftarrow \arg\max_{\mathcal{A}^2 : a_1 \in \mathcal{N}^{a_2}} \texttt{pr\_merge}(a_1, a_2)$
7:     **if** $|\mathcal{A}| \leq \hat{A} \wedge \texttt{pr\_merge}(a_1, a_2) \leq 0$ **then**
8:         **break**
9:     $a_1 \leftarrow a_1 \cup a_2$
10:     $\mathcal{A} \leftarrow \mathcal{A} \setminus \{a_2\}$
11: **return** $\mathcal{A}$

It is important to stress that evaluating the profit $\texttt{pr\_merge}$ does not necessarily mean computing the full performance function (III.4). Indeed, we can resort to simpler proxy functions, as detailed in Section IV-C.

2) The grow *approach:* The merge approach above is very simple; indeed, we perform a single operation – merge two areas – until the termination condition is reached. Simplicity is, in general, a good thing; however, some scenarios may call for a higher level of flexibility. In the following, we present an alternate approach, called *grow*.

We select the cell $c^\star$ that is best suited for a new area in line 3, and create a new area containing only this cell (line 5). Next, we grow the newly created area by selecting (line 7) a cell c' to add. c' is the cell, among the ones neighboring with area a, that is most profitable to add. If the profit is negative, then there are no more cells we can add to a (line 8), and we add a to the set $\mathcal{A}$ and move on creating the next area.

Algorithm 3 Grow approach

Require: $\mathcal{C}$
1: $\mathcal{A} \leftarrow \emptyset$
2: **while** True **do**
3:     $c^\star \leftarrow \arg\max_{c \in \mathcal{C}} \texttt{pr\_create}(c)$
4:     **if** $|\mathcal{A}| \leq \hat{A} \wedge \texttt{pr\_create}(c^\star) > 0$ **then**
5:         $a \leftarrow \{c^\star\}$
6:         **while** $|a| \leq |\mathcal{C}|$ **do**
7:             $c' \leftarrow \arg\max_{c \in \mathcal{C} : c \in N_a} \texttt{pr\_add}(c, a)$
8:             **if** $\texttt{pr\_add}(c, a) \leq 0$ **then break**
9:             $a \leftarrow a \cup \{c'\}$
10:         $\mathcal{A} \leftarrow \mathcal{A} \cup \{a\}$
11:     **else break**
12: **return** $\mathcal{A}$

This approach is more complex than the merge one, because of the two-phase structure of each step. However, with such a complexity comes a better flexibility, e.g., in defining cell-to-area assignments where areas overlap and there are cells not included in any area.

Similar to the previous case, notice that we have not given a definition of the $\texttt{pr\_add}$ and $\texttt{pr\_create}$ profit metrics. Different metrics can be adopted while pursuing different trade-offs between effectiveness and efficiency, as explained next.

C. Profit metrics

Profit metrics are evaluated very often during the execution of those algorithms; therefore, it is of paramount importance that they can be computed efficiently. However, such metrics must also represent a good proxy of the performance metric in (III.4).

There are two fundamental ways of defining profit. One is considering all such aspects as interference and propagation, and this essentially means computing (III.4) every time. The other is focusing on content demand, with the assumption that it is the main factor to account for in order to maximize performance.

1) Demand-based profit: Content demand π_m^c, i.e., the number of users in a cell interested in a certain content, is obviously the main factor to account for when taking such decision as creating or merging areas. For the sake of simplicity, we may decide to make it the *sole* factor to look at.

The $\texttt{pr\_merge}$ function used in line 6 of Algorithm 2 is thus defined as follows:

$$\texttt{pr\_merge}(a_1, a_2) = \frac{1}{U^{a_1} + U^{a_2}} \max_{m \in \mathcal{M}} \left(\sum_{c \in a_1 \cup a_2} \pi_m^c \right) \tag{IV.1}$$

where U^a denotes the number of users in area a. Equation (IV.1) says that we seek to merge those areas with a very strong interest in the same content (as opposed to a weaker interest for different content).
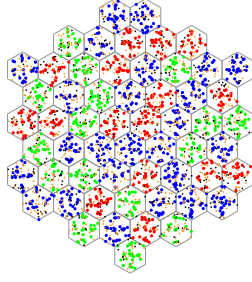
Fig. 1. Topology and demand: red, green and blue points correspond to *streaming* content, orange dots to the *update* one.

Fig. 2. Performance improvement (a) and area size (b) as functions of the number of areas, for different approaches and profit metrics.

Similarly, the `pr_create` function used in line 3 of Algorithm 3 is:

$$\texttt{pr\_create}(c) = \max_{m \in \mathcal{M}} \pi_m^c. \qquad \text{(IV.2)}$$

In (IV.2), we simply select the most popular content in cell c. Therefore, we tend to create new areas in those cells where there is a clearly prominent content.

Finally, the `pr_add` function used in line 7 accounts for how popular the content μ_a, broadcasted in area a, is in cell c, as shown in (IV.3):

$$\texttt{pr\_add}(c, a) = \pi_{\mu_a}^c. \qquad \text{(IV.3)}$$

Using the definitions above implies that side effects such as interference are neglected, but has a clear performance advantage. Content demand and interest are known a priori, and are not influenced by our decisions. Therefore, identifying and evaluating the possible actions is remarkably simple – and, thus, fast.

2) Holistic profits: The opposite approach consists in considering all the consequences of, say, adding a cell c to an area a, i.e., in accounting not only for the popularity of the content in the cell, but also for how the performance of other users, e.g., unicast ones, is affected.

This means to proceed as follows:

1) taking an action, e.g., adding cell c to area a or merging two areas a_1, a_2, and updating the α-values accordingly;
2) running Algorithm 1 on the resulting cell-to-area assignment;
3) recomputing the global score through (III.4), as explained in Section III.

V. RESULTS

Here, we first describe the network and traffic scenario that we used to derive performance results, then we present a comparison among the approaches introduced above.

A. Reference scenario

We evaluate our algorithms in a large-scale scenario typically used for 3GPP evaluation [11]. The scenario comprises a service area of 12.34 km^2, covered by 57 cells deployed at 19 tri-sectorial sites. There are a total of 3420 users, uniformly distributed under the cell coverage areas. Content is available as either *update* or *streaming*. The former, available as a single
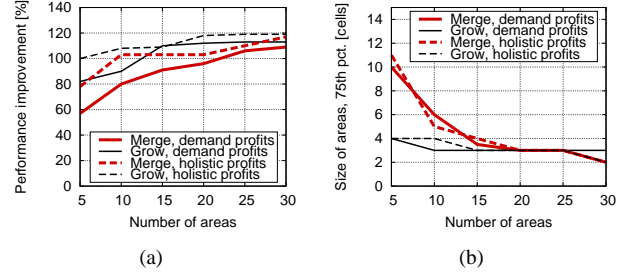
item, could represent a local map update, and it is less resource demanding. The latter could be seen as news clips streamed to users, and it is obviously more resource-demanding; we will consider three different items of streaming type.

We focus on a single time period, and assume that each user is interested in exactly one broadcast content item, as follows:

- with 20% probability, the user requests the *update* item;
- with 80% probability, the user requests one among *streaming1*, *streaming2* or *streaming3* items.

Furthermore, streaming items are location dependent, i.e., in each cell users may be interested in only one of the three item. The *update* content, instead, is requested throughout the whole topology – but with lower probability.

There are a total of $R = 500$ resources available per frame, each corresponding to an RB in LTE. At most 60% of such resources, i.e., $r = 300$, can be used for broadcast. The minimum amount of resources needed to broadcast a content is $\rho_m^c = 120$ for *streaming* content, and $\rho_m^c = 80$ for the *update* content. Topology and demand are summarized in Fig. 1.

B. Performance of the grow and merge approaches

The first thing we are interested in is the effectiveness of the two approaches we described in Section IV, i.e., *grow* and *merge*. We take as a reference the performance, computed through (III.4), when broadcasting is disabled, i.e., $\alpha_a^c = 0, \forall c, a$. Then, we measure how much such a performance is improved by enabling broadcasting, and using either approach
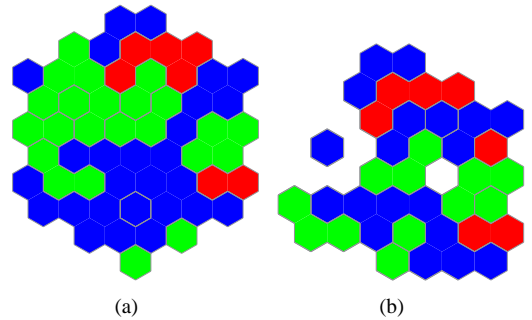

Fig. 3. Solutions yielded by the *merge* (a) and *grow* (b) approaches, for a maximum of 10 areas.

to form the area. The content to broadcast in each area is always chosen through Algorithm 1.

We vary the number of allowed areas, $\hat{A}$, between 5 and 30. These numbers are significantly smaller than the limit of 256 areas in the standard since our topology is much smaller than typical LTE geographical areas, which can span over hundreds of square kilometers.

Fig. 2(a) shows the performance improvement (with respect to the no-broadcast case) we obtain with the different approach and profit metrics.

A first observation we can make concerns the influence of the number maximum of areas: the bigger it is, the better the performance. This is expected; intuitively, more areas entail more flexibility.

Moving to the approaches, we observe that the *grow* approach consistently performs better than the *merge* one. As expected, enjoying a higher level of flexibility pays off. Less obviously, moving from the demand-based profit metric to the holistic one translates into a significant performance improvement only for the *merge* approach, and only when the limit on the number of areas to create is very tight.

Now, we have to determine why the *grow* approach performs better. Fig. 2(b) gives us an answer: it forms smaller areas. Recall the discussion in Section IV about the difference between the two approaches: with the *merge* approach, we are bound to put every cell in (exactly) one area. This may not sound like a bad idea in our scenario; after all, we have a significant demand for broadcast-able content throughout the whole topology.

It turns out, instead, that insisting to have all cells assigned to an area severely impairs performance. We can get an idea of why this happens by looking at Fig. 3. As expected, the *merge* approach yields a solution where all cells belong to an area (Fig. 3(a)). By comparing Fig. 3(a) to the demand in Fig. 1, however, we can see that many cells are in areas that broadcast a content that nobody wants.

Compare now the solution yielded by the *grow* approach, in Fig. 3(b). The first thing that strikes our attention is that many cells do not belong to any area. Looking more carefully, we can see that this typically happens with cells surrounded by neighbors with different demand (look, e.g., at the "hole" towards the center of the topology or the "island" on the left). These cells are never selected during the *grow* process (Algorithm 3), and therefore all the resources therein are used for unicasting. As we can see from Fig. 2(a), this is more convenient than broadcasting content with low popularity.

Summary. We can thus draw three main conclusions from our performance evaluation. First, the *grow* approach outperforms the *merge* one, owing to its higher level of flexibility. Second, such a flexibility is sufficient to compensate the usage of a simpler profit metric, namely, interest-based. Third, such a difference in performance is mostly due to the tendency of the *merge* approach to assign each cell to an area, at the cost of broadcasting uninteresting content.

VI. CONCLUSIONS

We have considered the broadcasting features in LTE. Specifically, we addressed the problems of (i) forming the areas, i.e., decide which cell(s) belong to each area, and (ii) deciding which content to broadcast in each cell. We decoupled the LTE broadcasting problems of forming the areas and choosing the content to broadcast. We presented a simple and straightforward strategy for the last problem, and two approaches, presenting different levels of complexity and flexibility, for the first. Additionally, we described two ways of assessing the profitability of possible assignments: accounting for content demand alone, or adding interference issues to the picture. By evaluating our system in a large-scale, real-world scenario, we found that selecting the most flexible approach makes it possible to use the simplest profit metric, thus being able to (re)schedule the content to broadcast on a very fine time granularity. We also investigated the reason for the difference in performance between the two approaches, and found that trying to assign all cells to an area is harmful to the overall performance.

REFERENCES

[1] F. Malandrino, C. Casetti, C.-F. Chiasserini, Z. Limani, "Fast Resource Scheduling in HetNets with D2D Support," *IEEE INFOCOM*, Toronto, Canada, 2014.

[2] S. Deb, P. Monogioudis, J. Miernik, J. P. Seymour, "Algorithms for Enhanced Inter-Cell Interference Coordination (eICIC) in LTE HetNets," *IEEE/ACM Transactions on Networking*, Vol. 22, No. 1, 2013.

[3] K. Fitchard, "Can LTE-broadcast Dam the Mobile Video Deluge?" http://gigaom.com/2013/01/10/can-lte-broadcast-dam-the-mobile-video-deluge/ [Accessed Feb. 2014]

[4] L. Rong, O. Ben Haddada, S. Elayoubi, "Analytical Analysis of the Coverage of a MBSFN OFDMA Network," *IEEE GLOBECOM*, New Orleans, LA, USA, 2008.

[5] A. Alexiou, C. Bouras, V. Kokkinos, A. Papazois, G. Tsichritzis, "Efficient MCS Selection for MBSFN Transmissions over LTE Networks," *IFIP Wireless Days*, Venice, Italy, 2010.

[6] R. Radhakrishnan, B. Tirouvengadam, A. Nayak, "Channel Quality-based AMC and Smart Scheduling Scheme for SVC Video Transmission in LTE Networks," *IEEE ICC*, Ottawa, Canada, 2012.

[7] T. Lohmar, M. Slssingar, V. Kenehan, S. Puustinen, "Delivering Content with LTE Broadcast," *Ericsson Review*, Feb. 11, 2013.

[8] 3GPP TS 22.146, "Multimedia Broadcast/Multicast Service; Stage 1 (Release 6)", V6.7.0, Mar. 2006.

[9] 3GPP TS 36.300, "Evolved Universal Terrestrial Radio Access (E-UTRA) and Evolved Universal Terrestrial Radio Access (E-UTRAN); Overall Description (Release 9)", V9.9.0, Dec. 2011.

[10] C.F. Chiasserini, "On the Concept of Distributed Digital Signal Processing in Wireless Sensor Networks," *IEEE MILCOM*, Anaheim, CA, USA, 2002.

[11] 3GPP Technical Report 36.814, "Further Advancements for E-UTRA Physical Layer Aspects," 2010.