Backhaul Design and Controller Placement for Cooperative Mobile Access Networks

Thorsten Biermann*, Luca Scalia*, Jörg Widmer†, and Holger Karl+
*DOCOMO Euro-Labs, Munich, Germany, †Institute IMDEA Networks, Madrid, Spain, +University of Paderborn, Germany
{biermann, scalia}@docomolab-euro.com, widmer@imdea.org, holger.karl@upb.de

***Abstract*—Exploiting base station cooperation in wireless mobile access networks leads to benefits in wireless transmission capacity, inter-cell interference management, and cell edge user experience. The clustering of cooperative Base Station (BS) sets, necessary for achieving the desired wireless performance, poses several challenges in the backhaul architecture design. This paper addresses the problem of placing and connecting controller/ processing nodes within the backhaul infrastructure, which coordinate and/or process signals of cooperating base stations. We formulated a Mixed Integer Linear Programm (MILP) for this problem and a heuristic algorithm that approximates the optimal solution. While the heuristic's solution quality is close to the optimum, the runtime and memory requirements are multiple orders of magnitude lower compared to solving the MILP. This advantage allows to use the proposed heuristic either for backhaul/core network pre-planning or for on-the-fly network reconfiguration during ongoing mobile network operation.**

I. Introduction

Cooperation in cellular mobile access networks shows high benefits in terms of wireless transmission capacity, inter-cell interference management, and cell edge user experience. Several techniques have been proposed so far (joint precoding and decoding, inter-cell coordination, etc.), each of them having different merits.

There has been a lot of research in the area of implementing and evaluating the performance of different cooperation techniques, like joint signal processing [1]–[3] and inter-cell coordination [4], [5], from the wireless point of view. The results show how to integrate the cooperation techniques into the wireless systems, demonstrate the possible performance gains, and partly scratch the surface on how to design an overall network architecture that incorporates cooperation techniques. Other work covers (i) the synchronization of the components involved in the cooperation schemes [6], (ii) the number of BSs that need to cooperate to achieve the desired gains [7], (iii) the management of inter-cluster interference [8], e.g., by introducing overlapping cluster configurations, and (iv) the efficient collection of Channel State Information (CSI) information [9].

The common denominator of most cooperation techniques are central controller/processing nodes within the core/ backhaul network of the mobile operator. Although the impact of a limited wireline network has been studied [10], the problem of *where* to optimally place these nodes inside the network and *how* to connect them to the BSs is not addressed. For this, requirements of the cooperation techniques and properties of the wireline network both have to be taken into account. Otherwise, the wireless performance decreases or cooperation even becomes impossible.

This work was done while Jörg Widmer was with DOCOMO Euro-Labs.

In the following, we first formulate the problem that is addressed in this paper in detail in Sec. II before we present a MILP that solves this problem optimally (Sec. III). Thereafter, we propose a heuristic algorithm in Sec. IV that approximates the optimal solution while reducing the runtime and memory requirements. Sec. V presents an evaluation of both approaches. Finally, we discuss application scenarios of the heuristic (Sec. VI) before concluding our work in Sec. VII.

II. Problem description

Clustering cooperative BS sets, necessary for achieving the desired wireless performance, poses several challenges in the backhaul architecture design. Depending on the used cooperation scheme, a central controller and/or processing node and a set of cooperating BSs need to exchange signaling and sometimes user data traffic.

The placement of these controller/processing nodes and their connection to the cooperating BSs play a crucial role in the overall backhaul architecture design. System parameters like propagation delay in the wired part of the network, required link capacity for transporting user data and control information, synchronization between cooperative BSs, Capital Expense (CAPEX) and Operating Expense (OPEX) of the backhaul architecture need to be taken into account.

This leads to the following problem for designing backhaul networks for cooperative, cellular mobile access networks. Given a set of BSs and their (potential) interconnections, we need to know how to optimally position controller/processing nodes in the network and how to assign BSs to these nodes.

To solve this network design problem, we use the parameters in Tab. I to define the input scenario. These parameters define a graph that has a vertex for each BS $b \in B$ and an edge for each link $l \in L$ between the BSs. All vertices and edges are augmented with properties, like capacity and latency, that are important for the network design process. Additionally, there are four global properties (cs_{min}, cs_{max}, t_{proc}, and t_{max}) that are independent of BSs or links.

III. Optimal Solution

This section introduces a Mixed Integer Linear Programm (MILP) that optimally solves the network design problem

TABLE I: Input scenario parameters

B	Set of BSs
L	Set of links, where $L \subseteq B \times B$
b_costF_u	Fixed costs for setting up a controller at BS $u \in B$
b_costC_u	Costs per controlled BS for a controller at BS $u \in B$
b_cap_u	Required capacity (e.g., bandwidth/data rate) at BS $u \in B$
$\text{l\_cap}_{u,v}$	Capacity (e.g., bandwidth/data rate) of link $(u, v) \in L$
$\text{l\_t}_{u,v}$	Latency of link $(u, v) \in L$
$\text{l\_cost}_{u,v}$	Costs of link $(u, v) \in L$
cs_{min}	Minimum required cluster size
cs_{max}	Maximum allowed cluster size
t_{proc}	Processing time at controller
t_{max}	Maximum total round trip time from BS to controller

introduced in Sec. II. Based on the input, the MILP calculates optimal positions for the controller/processing nodes and assigns BSs to these nodes while minimizing costs. This assignment leads to the optimal interconnection of all BSs.

In the following, we will use monetary costs as optimization goal. This can easily be adapted to a different cost metric like energy consumption.

The MILP uses the parameters in Tab. I as input and uses the variables shown in Tab. II to optimize the metric.

TABLE II: Optimization variables

$\text{l\_act}_{u,v}$	Determines whether link $(u, v) \in L$ is active, $\text{l\_act}_{u,v} \in \{0, 1\}$
b_act_u	Determines whether a controller/ processing node is colocated at BS $u \in B$, $\text{b\_act}_u \in \{0, 1\}$
$\text{f}_{s,d,u,v}$	Determines whether a data flow from a controller/processing node at $s \in B$ goes to $d \in B$ over link $(u, v) \in L$, $\text{f}_{s,d,u,v} \in \{0, 1\}$
b_actCostF_u	Actual fixed controller/processor costs at $u \in B$, $\text{b\_actCostF}_u \geq 0$
b_actCostC_u	Actual controller/processor costs at $u \in B$ per each controlled BS, $\text{b\_actCostC}_u \geq 0$
$\text{l\_actCost}_{u,v}$	Actual costs for link $(u, v) \in L$, $\text{l\_actCost}_{u,v} \geq 0$

Setting $\text{l\_act}_{u,v}$ to 1 means that the link (u, v) transports some flow, i.e., there is at least one $\text{f}_{s,d,u,v}$ set to 1. Similar to this, if b_act_u is set to 1, a controller/processing function is colocated at BS u. At the moment, our MILP supports to colocate these functions at a BS; they cannot be positioned arbitrarily in the network.

The three variables b_actCostF_u, b_actCostC_u, and $\text{l\_actCost}_{u,v}$ contain the actual costs for controllers and links. These costs depend on whether a controller/processing node is actually colocated at a BS and on whether a link is active, i.e., used to connect clustered BSs, or not.

To get the total link and controller/processing node costs for a network configuration, we need to sum up the individual costs. This is done in Eq. (1) and Eq. (2), respectively.

$$\text{l\_cost}_{\text{total}} = \sum_{(u,v)\in L} \text{l\_actCost}_{u,v} \tag{1}$$

$$\text{b\_cost}_{\text{total}} = \sum_{u\in B} \text{b\_actCostF}_u + \text{b\_actCostC}_u \tag{2}$$

The goal is to minimize the total costs while taking into account the constraints of the applied wireless cooperation scheme, defined by the parameters in Tab. I. The following MILP achieves that:

$$\text{min.} \quad \text{b\_cost}_{\text{total}} + \text{l\_cost}_{\text{total}} \tag{3}$$

$$\text{s. t.} \quad \sum_{s\in B,(u,d)\in L} \text{f}_{s,d,u,d} = 1, \ \forall d \in B, \tag{4}$$

$$\text{s. t.} \quad \sum_{s\in B,(s,u)\in L} \text{f}_{s,d,s,u} = 1, \ \forall d \in B, \tag{5}$$

$$\text{s. t.} \quad \sum_{(v,u)\in L} \text{f}_{s,d,v,u} = \sum_{(u,w)\in L} \text{f}_{s,d,u,w}, \quad \forall u \in B, (s, d) \in B \times B, u \neq s, u \neq d, \tag{6}$$

$$\text{s. t.} \quad \text{f}_{s,d,u,v} = 0, \forall (s, d, u, v) \in B \times B \times B \times B, \quad s \neq d, u = v, \tag{7}$$

$$\text{s. t.} \quad \text{l\_act}_{u,v} \cdot M \geq \sum_{(s,d)\in B\times B} \text{f}_{s,d,u,v}, \ \forall (u, v) \in L, \tag{8}$$

$$\text{s. t.} \quad \text{b\_act}_s \cdot M \geq \sum_{d\in B,(s,v)\in L} \text{f}_{s,d,s,v}, \ \forall s \in B, \tag{9}$$

$$\text{s. t.} \quad \sum_{(d,v)\in B\times B,(c,v)\in L} \text{f}_{c,d,c,v} \geq cs_{\text{min}} \cdot \text{b\_act}_c, \quad \forall c \in B, \tag{10}$$

$$\text{s. t.} \quad \sum_{(d,v)\in B\times B,(c,v)\in L} \text{f}_{c,d,c,v} \leq cs_{\text{max}}, \ \forall c \in B, \tag{11}$$

$$\text{s. t.} \quad \sum_{(s,d)\in B\times B} \text{f}_{s,d,u,v} \cdot \text{b\_cap}_d \leq \text{l\_cap}_{u,v}, \quad \forall (u, v) \in L, \tag{12}$$

$$\text{s. t.} \quad \sum_{(s,d)\in B\times B} \text{f}_{s,d,u,v} \cdot \text{b\_cap}_d \leq \text{l\_cap}_{v,u}, \quad \forall (v, u) \in L, \tag{13}$$

$$\text{s. t.} \quad \sum_{(u,v)\in L} \text{f}_{s,d,u,v} \cdot (\text{l\_t}_{u,v} + \text{l\_t}_{v,u}) \leq t_{\text{max}} - t_{\text{proc}}, \quad \forall (s, d) \in B \times B, \tag{14}$$

$$\text{s. t.} \quad \text{b\_actCostF}_c = \text{b\_act}_c \cdot \text{b\_costF}_c, \ \forall c \in B, \tag{15}$$

$$\text{s. t.} \quad \text{b\_actCostC}_c = \sum_{d\in B,(c,v)\in L} \text{f}_{c,d,c,v} \cdot \text{b\_costC}_c, \quad \forall c \in B, \tag{16}$$

$$\text{s. t.} \quad \text{l\_actCost}_{u,v} = \text{l\_act}_{u,v} \cdot \text{l\_cost}_{u,v}, \ \forall (u, v) \in L, \tag{17}$$

The first constraint in Eq. (4) ensures that each BS is assigned to exactly one controller. This is done by fixing the number of flows that end at each BS to 1. The second constraint, shown in Eq. (5), guarantees that each flow that ends at a BS also has a start BS – the controller. Eq. (6) contains the third constraint, which is necessary to create the flows in the network. It guarantees the flow balance, i.e., whenever a flow enters a node it also has to leave it again, except the node is the flow's source or destination. Finally, Eq. (7) forbids local loops for each flow.

After having created the flows between the controllers and the BSs, the helper constraint in Eq. (8) activates all links in the network that are required to transport one of the flows. This constraint uses a "big-M constant" to achieve this. Similarly, Eq. (9) activates the controller functionality at each BS that is source of at least one flow.

As many cooperation schemes, like joint precoding, require a minimum amount of BSs being jointly controlled to achieve the desired gains, the constraint in Eq. (10) requires at least cs_{min} BSs to be connected to the same controller. At the same time, Eq. (11) limits the maximum cluster size to cs_{max}. This might be useful in case the controller capacities are limited and hence clusters cannot exceed a certain size.

The constraints in Eq. (12) and Eq. (13) ensure that the link capacities are not exceeded in the downlink (from controller to the BS) and in the uplink, respectively.

The next constraint in Eq. (14) ensures that the round-trip delay between a controller and a BS is below an upper bound. This bound is calculated by subtracting the required processing time t_{proc} at the controller from the maximum allowed delay t_{max}. Such a constraint is important, e.g., for joint precoding. Here, the encoded data must be sent while the CSI, based on which the encoding was done, is still valid.

Finally, the last three constraints calculate the link and controller costs for the resulting network. Eq. (15) and Eq. (16) define the fixed and dynamic controller costs. Eq. (17) sets the costs for each link depending on whether it is active or not.

The presented MILP returns the *optimal* solution for the problem in Sec. II. Finding such a solution, however, is NP-hard and requires a long solver runtime.

IV. Heuristic Approach

To overcome the MILP's runtime problem, we propose a heuristic to approximate the optimal solution. The heuristic takes the same input as the MILP (i.e., a property graph containing the BSs and potential interconnections) and produces the same kind of output (locations of controllers/processors and optimized interconnections between these components).

We have chosen a greedy approach based on a modified Breadth First Search (BFS) algorithm to build clusters of BSs. The resulting heuristic is summarized in Alg. 1.

The method CONFIGURECOOPNETWORK expects the input parameter G_{in}. This is the input graph of BSs and potential interconnections. The algorithm first initializes the output graph G_{out}. After this, it traverses a loop where in each iteration one new cluster is created and added to G_{out}. Furthermore, BSs that have been assigned to a cluster are removed from G_{in}.

Algorithm 1 CONFIGURECOOPNETWORK$(G_{\text{in}}(B, L))$

1: $G_{\text{out}} = \emptyset$ // initialize output graph with empty set
2: **while** $|B| > 0$ **do**
3: $\quad s =$ CHOOSECONTROLLER(G_{in})
4: $\quad G_{\text{newClust}} =$ BFS'(G_{in}, s) // calculate new cluster
5: $\quad G_{\text{out}} = G_{\text{out}} \cup G_{\text{newClust}}$ // add new cluster to G_{out}
6: $\quad G_{\text{in}} = G_{\text{in}} \setminus G_{\text{newClust}}$ // remove new cluster from G_{in}
7: **end while**
8: **return** G_{out}

At the beginning of each iteration, one of the BSs in G_{in} is selected that will act as a cluster controller. Starting from this new controller s, the BFS' algorithm builds the cluster around s and returns the resulting cluster graph G_{newClust}. This graph contains all BSs that are member of the cluster and links that connect these BSs to the controller s.

The way how to choose s from G_{in} plays an important role for the quality of the heuristic's output. The method CHOOSECONTROLLER selects a BS from G_{in} such that its distance to the border of G_{in} equals the expected radius of the cluster that is currently created.

The size (and hence the radius) of the resulting cluster depends on different things: the physical properties of the involved BSs and their interconnections, and the constraints that are imposed by the wireless cooperation technique.

In the following, we give an example how this can be done for joint precoding as cooperation technique (Alg. 2).

Algorithm 2 CHOOSECONTROLLER$(G(B, L))$

1: $cr_{\text{lat}} = (t_{\text{max}} - t_{\text{proc}}) \,/\, \left(2 \cdot \sum_{(u,v)\in L} \text{l\_t}_{u,v}/|L|\right)$
2: $cr_{\text{cap}} = \left(\sum_{(u,v)\in L} \text{l\_cap}_{u,v}/|L|\right) \,/\, \left(\sum_{u\in B} \text{b\_cap}_{u}/|B|\right)$
3: $cr_{\text{overall}} = \text{MIN}(cr_{\text{lat}}, cr_{\text{cap}})$ // expected cluster radius (in hops)
4: $s =$ GETNODENHOPSFROMBORDER(G, cr_{overall})
5: **return** s

Joint precoding has two major constraints that have to be fulfilled. The first one is that the latency from measuring the CSI until the point in time where the jointly encoded data is sent must be below an upper bound, e.g., 1 ms for Long Term Evolution (LTE). Hence, the processing delay at the controller and the Round Trip Time (RTT) from the controller to the controlled BSs must not exceed this limit. Accordingly, the expected cluster radius regarding the latency cr_{lat} can be calculated as shown in the first line of Alg. 2. The parameters t_{max} and t_{proc} contain the maximum allowed time span from measuring the CSI to the actual sending of the data, and the processing delay at the controller, respectively (as defined in Tab. I). Their difference is divided by the mean link latency.

The second limitation for joint precoding is that the links must have enough capacity to transport the signals from the controller to the BSs from where they are sent. So the amount of bandwidth that is required at each BS and the link capacities from the controller to the BSs (possibly involving multiple hops) limit the maximum possible cluster size, too. This is taken into account in line 2 of Alg. 2. There, the expected cluster radius depending on the capacity properties of the input

graph cr_{cap} is calculated. The radius depends on the mean link capacity and the mean capacity requirement of the BSs.

As a result, the overall expected cluster radius cr_{overall} is the minimum of cr_{lat} and cr_{cap}. Using cr_{overall} as input, the method GETNODENHOPSFROMBORDER returns a BS from the input graph that is exactly cr_{overall} hops away from the graph's border. This BS is the new controller.

Calculating the expected cluster radius of course changes for different kinds of wireless cooperation as they might have different constraints that have to be fulfilled.

Now that the heuristic has selected a new controller s from the input graph G_{in} using CHOOSECONTROLLER, a modified BFS is started with s as start node. Compared to a standard BFS algorithm, whenever the BFS' algorithm traverses a link and finds a new BS x in G_{in}, it checks whether x can be added to the current cluster without violating any constraint. If this is possible, x is added to the current cluster and to the BFS's queue of BSs to continue the search from later on. If adding x to the cluster is not possible, the search algorithm does not continue searching for new BSs on this path of the search tree. As soon as the BFS' algorithm terminates, a new cluster has been created around the controller s.

After the BFS' has terminated, the new cluster (represented by G_{newClust}) is added to the output graph G_{out} and removed from the input graph G_{in}. The loop continues with the next iteration until all BSs have been assigned to a cluster.

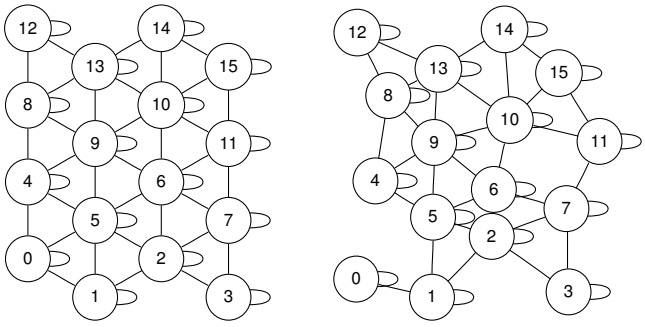
V. EVALUATION

This section covers the evaluation of solving the problem using the MILP described in Sec. III and approximating the optimal solution using our heuristic algorithm from Sec. IV. In the following figures, all confidence intervals have been calculated for a confidence level of 95 %.

The input scenario consists of a matrix of n BSs that form a square. We look at two different arrangements of the BSs. The first is a regular arrangement where all BSs have identical inter-BS distances of $\overline{s} = 1600 \cdot \sqrt{3}\,\text{m} \approx 2771\,m$ [11] and have potential interconnections to all neighboring BSs. The second type of arrangement is irregular and adds randomness to the BS positioning. Hereby, BSs still have a *mean* distance of $\overline{s}$ but positions are normally-distributed with standard deviation $\overline{s}/8$. BSs are interconnected if their distance is below $1.25 \cdot \overline{s}$. The reason for having irregular input scenarios is to check whether our heuristic suffers from arbitrarily choosing new neighbors during the BFS. Fig. 1 shows two examples of these two arrangement types for $n = 16$.

The interconnections are optical and have a latency of $s \cdot \frac{1.45}{c}$, where c is the speed of light and 1.45 the refraction index of the fiber. This corresponds to a typical Single-Mode Fiber (SMF) setup. We further assume a Radio over Fiber (RoF) setup from the controllers to the BSs.[1] The RoF capacity of each link is 3000 MHz, which corresponds to standard RoF equipment. Every BSs consists of 3 sectors that have 4 antennas each, operated at a bandwidth of 100 MHz, which

[1]Note that this only affects the evaluation. Our proposed methods are generic enough to support any technology.



(a) Regular arrangement (b) Irregular arrangement

Fig. 1: Example input scenarios for $n = 16$

corresponds to a future wide band wireless system. Tab. III summarizes the input parameters.

TABLE III: Chosen input parameters

Parameter	Value
C	$\{1..n\}$
L	$\{(u, v) \in C \times C\}$
b_costF_u	3
b_costC_u	0.5
b_cap$_u$	$3 \cdot 4 \cdot 100\,\text{MHz} = 1200\,\text{MHz}$
l_cap$_{u,v}$	3000 MHz if $u \neq v$, else ∞
l_t$_{u,v}$	$s \cdot 1.45/c \approx 13.4\,\mu\text{s}$ (on average)
l_cost$_{u,v}$	$s/\overline{s} = 1$ (on average)
cs_{min}	3
cs_{max}	50
t_{proc}	0.5 ms
t_{max}	1 ms

We did experiments for $n \in \{3^2, 4^2, 5^2, 6^2, 7^2\}$ on a dual-core machine (3.33 GHz per core, 4 GB RAM) while setting the optimality gap of the MILP solver to 0.1. This means that the returned solution is at maximum 10 % worse than the optimal solution. Such a high gap was required to achieve solver runtimes below one week for the larger input scenarios.

Compared to solving the problem optimally with the MILP, the heuristic method has a much shorter runtime and lower memory requirements. The difference is illustrated in Fig. 2 for regular (*reg*) and irregular (*irreg*) input scenarios. The figure shows two versions of our heuristic method to later demonstrate the advantage of choosing the start nodes for the BFS. The *simple* one simply picks the first node at the border of the input graph. The *mhop* heuristic, however, chooses new controller locations as proposed in Alg. 2.

The plot shows that while the MILP solver runtime increases up to 20 hours, the heuristic's runtime stays between 1.5 and 6 orders of magnitude below that (runtime always below 1 s).

To get an idea of the quality of the heuristics' output, Fig. 3 compares the output given by the MILP solver to the output of the two versions of our proposed heuristic. The resulting costs are the total costs for the returned network configuration,

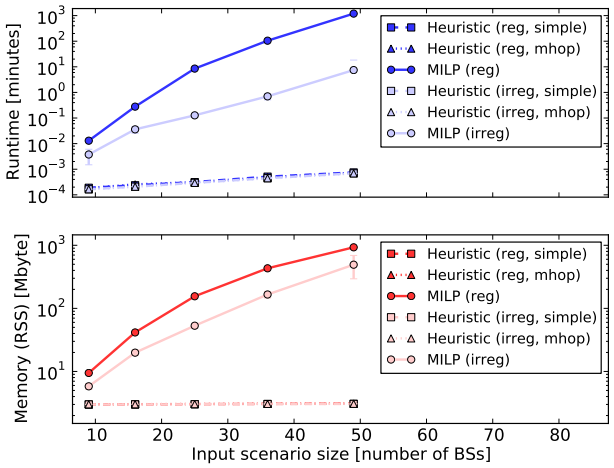
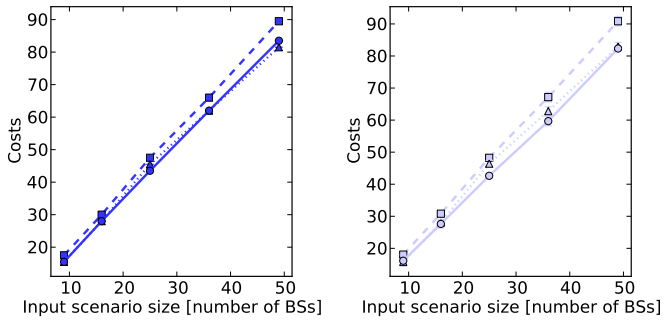


Fig. 2: Solver runtime and memory consumption depending on input scenario size

using the unit costs defined in Tab. III.



(a) Regular input scenario (b) Irregular input scenario

Fig. 3: Costs of the resulting network configuration depending on input scenario size. Please refer to the legend of Fig. 2.

The returned output of the heuristic is at maximum 5 % worse than the output provided by the MILP solver. Furthermore, the difference between the simple and mhop heuristic shows that our proposed way of choosing a new controller has a major influence, especially for irregular input scenarios, i.e., where BSs are distributed randomly in the plane and where links have different properties in terms of capacity and latency.

Note that in Fig. 3a, the heuristic (mhop) even produces lower costs than the MILP. This is possible as we set the optimality gap of the MILP solver to 10 %. The result of the heuristic, however, is better than that.

VI. Application Scenarios

The evaluation has shown that our heuristic algorithm provides an output quality that is close to the optimal solution. The runtime and the memory requirements, however, are significantly lower compared to the ones of a MILP solver. These advantages allows to use our method in two application scenarios: offline and online network configuration.

In the offline case, the heuristic approach is applied *before* deploying the actual backhaul/core network infrastructure. This corresponds to a network planning procedure when a mobile operator intends to augment its existing network with base station cooperation.

In the online case, the cooperative network is already deployed. Here, it can be required to, e.g., re-assign certain base stations to other cooperation clusters to react on changes in the network, like link failures or varying load. This information about the network is collected during its operation and provided as input data to the heuristic, which determines what to change in the network to improve the operation.

VII. Conclusions

We proposed two methods to solve the problem of how to position controller/processor nodes in a cooperative mobile access network and how to assign BSs to them. The MILP returns optimal solutions but is only feasible for small input scenarios. The heuristic produces solutions that are only 5 % worse than the MILP's output but has a runtime and memory consumption that is magnitudes lower compared to the MILP.

The heuristic allows to calculate quasi-optimal solutions for large input scenarios and can be applied during the runtime of a network to reconfigure it on the fly, thanks to the low resource requirements. Next steps are to extend the heuristic to position controllers independently of the BS locations, to take into account shared resources in tree or ring topologies, and to support multiple overlapping clusters per BS.

References

[1] V. Jungnickel, L. Thiele, T. Wirth, T. Haustein, S. Schiffermüller, A. Forck, S. Wahls, S. Jaeckel, S. Schubert, H. Gäbler, and Others, "Coordinated multipoint trials in the downlink," in *Proc. IEEE Broadband Wireless Access Workshop (BWAWS)*, Nov. 2009.

[2] V. Jungnickel, M. Schellmann, L. Thiele, T. Wirth, T. Haustein, O. Koch, W. Zirwas, and E. Schulz, "Interference-aware scheduling in the multiuser MIMO-OFDM downlink," *IEEE Communications Magazine*, vol. 47, no. 6, pp. 56–66, June 2009.

[3] S. Venkatesan, H. Huang, A. Lozano, and R. Valenzuela, "A WiMAX-based implementation of network MIMO for indoor wireless systems," *EURASIP Journal on Advances in Signal Processing*, 2009.

[4] W. Choi and J. G. Andrews, "The capacity gain from intercell scheduling in multi-antenna systems," *IEEE Transactions on Wireless Communications*, vol. 7, no. 2, pp. 714–725, Feb. 2008.

[5] J. G. Andrews, A. Ghosh, and R. W. Heath, "Networked MIMO with clustered linear precoding," *IEEE Transactions on Wireless Communications*, vol. 8, no. 4, pp. 1910–1921, Apr. 2009.

[6] V. Jungnickel, T. Wirth, M. Schellmann, T. Haustein, and W. Zirwas, "Synchronization of cooperative base stations," in *Proc. IEEE Int. Symp. on Wireless Communication Systems (ISWCS)*, 2008, pp. 329–334.

[7] J. Hoydis, M. Kobayashi, and M. Debbah, "On the optimal number of cooperative base stations in network MIMO systems," *Arxiv preprint arXiv:1003.0332*, 2010.

[8] G. Caire, S. A. Ramprashad, and H. C. Papadopoulos, "Rethinking network MIMO: Cost of CSIT, performance analysis, and architecture comparisons," in *Proc. IEEE Inf. Theory and Appl. WS (ITA)*, Jan. 2010.

[9] L. Thiele, M. Schellmann, S. Schiffermüller, V. Jungnickel, and W. Zirwas, "Multi-cell channel estimation using virtual pilots," in *Proc. IEEE Vehicular Technology Conference (VTC)*, May 2008, pp. 1211–1215.

[10] P. Marsch and G. Fettweis, "On downlink network MIMO under a constrained backhaul and imperfect channel knowledge," in *Proc. IEEE Global Comm. Conference (GLOBECOM)*, 2009, pp. 5059–5064.

[11] A. Benjebbour, M. Shirakabe, Y. Ohwatari, J. Hagiwara, and T. Ohya, "Evaluation of user throughput for MU-MIMO coordinated wireless networks," in *Proc. IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, Sept. 2008.