Trade-off between Latency and Coverage in Cooperative Radio Access Networks

(Preliminary preprint)
Till Hohenberger, Matthias Herlich, and Holger Karl
tiho@mail.upb.de, matthias.herlich@upb.de, holger.karl@upb.de

Abstract—To cope with higher demands, future radio access networks will consist of densely deployed base stations (BSs). As the BS ranges in such deployments will often overlap, areas exists that could be served individually by one of several BSs or cooperatively by all members of a group of BSs (using techniques like CoMP). One BS typically serves several such areas.

Each BS has to decide, which area to serve when. This problem is compounded by the need to coordinate cooperative transmissions across several BSs. We model this as a queuing problem where each queue represents one particular area that is served by a specific combination of BSs; each BS has to decide which queue(s) to serve when. For three decision strategies, we derive a Markov model and solve it numerically to obtain steady-state solutions.

These solutions let us quantify average latency and trade-offs between latency and number of BSs per area for both direct and cooperative scenarios. We show that cooperation reduces the number of BSs needed to cover an area but that a latency price has to be payed.

Index Terms—Performance evaluation, Radio access networks

I. INTRODUCTION

Traditional radio access networks serve jobs, generated by users, by assigning them to a single base station (BS). Beginning with Long Term Evolution (LTE)-Advanced multiple BSs can cooperate to serve jobs. Coordinated Multi-Point (CoMP) is one possible technique, to implement cooperative transmission [1]. This increases the BSs’ coverage (as seen in Figure 1(a)) on the cost of higher average latency, since each BS has a higher workload.

As the coverage of some BSs overlap, these BSs have the option to select, which of them serves the shared area (Area 2 in Figure 1(a)). This enables to reduce the average latency, as BSs with low load can help out the others.

The covered areas are either served by (1) one single BS (Area 1 and 3 in Figure 1(a)), (2) two or more cooperating BSs, where each job is processed by all BSs, but not necessarily at the same time (Area 4) or (3) by two or more BSs, but each job is only served by one BS (Area 2). Each BS can select jobs from different (shared and unshared) areas to serve.

The example shown in Figure 1(a) can be transformed into a queuing system, as seen in Figure 1(b). In the queuing system, the BSs have to select which queue to serve next.

In this paper, we compare different strategies for the BSs to select areas to serve. The comparison is done using the average latency, i.e. the expected duration until a job is finished. We consider the strategies Random and Round-robin, as well as Longest Connected Queue in which each BS serves the queue with the most waiting jobs under all its connected queues. We also compare these strategies to a lower bound on latency, using classical $M/M/k$ queuing theory [2], [3]. In addition to assigning each job to a single BS we also determine the effects of cooperative transmissions. Using cooperation increases the coverage, allowing a more sparse placement of BSs, with up to $\approx 37.4\%$ less BSs [4], but has the drawback that cooperatively served jobs need to be processed by multiple BSs. As this increases the load of each BS, it also increases the average latency of the jobs by queuing effects. Our contribution is to quantify the trade-off between the sparser placement and
the increase of latency. Additionally, we compare different strategies to select areas to serve. This is done for scenarios with two BSs (as seen in Figure 1(a)) and three BSs.

An analytical analysis does not seem possible as even two-dimensional Markov chains need approximation to be solved [5]. In our case the problem gets even harder as the Markov chains have more dimensions. We, therefore, decided to derive the Markov chains’ steady state distribution numerically, unlike most related work, which uses simulations. The results give an overview about strategies to coordinate BSs efficiently by explicitly modeling queuing effects using Markov chains. Furthermore, we quantify the trade-offs between latency and coverage for the placement of BSs.

II. RELATED WORK

Couto da Silva et al. [6] determine the energy consumption of dense WLANs. They numerically compute the possible power saving for a two-BSs scenario without cooperation. Our paper enhances their work by adding cooperation and considering a three-BSs scenario. Our focus is the trade-off between coverage and latency that arises by varying the cell overlap.

McKeown et al. [7] show that the Longest Connected Queue (LCQ) strategy achieves 100% throughput for i.i.d. arrivals and fixed service times at an input queued switch. Although, the service times in our case are exponential distributed and our scenarios differ from an input queued switch, their proof justifies the choice of LCQ as one of our strategies.

Cycle Stealing considers two workstations, each with its own queue. One of the workstations (donor) is able to help the other workstation (beneficiary), by completing jobs out of the beneficiary’s queue. Osagami [5] analytically approximates Markov chains for this problem. The cycle stealing problem is a special case of our problem, as in our definition both workstations may help each other and, therefore, are able to switch roles.

Cooperation can be implemented using Coordinated Scheduling / Beamforming. The BSs coordinate their transmission to reduce interference and thereby maximize the throughput. As mentioned by Kishiyama et al. [1] this mainly improves the throughput at the cells’ edges. Beamforming as specified by LTE works with up to four antennas but need the BSs to transmit at the same time and therefore coordinate the transmission. We ignore the BSs’ communication between each other as well as the constraint to transmit at the same time.

In earlier work [4] we describe the impact of cooperation on the power consumption. We analytically show that cooperation supersedes some of the BSs which can be switched off to decrease consumed power. Our new work can be used to extend our old results by considering the energy consumption when trading off latency and coverage.

III. ABSTRACTIONS AND DEFINITIONS

We model a radio access network consisting of BSs and transmission jobs, submitted by users.

A. Definitions

This section introduces the basic definitions and assumptions we make for the analysis.

1) Base stations: Each BS covers the area of a disk and is able to serve jobs in its area. Note that this assumption is only necessary to determine the sizes (and arrival rates) of the areas and can easily be replaced with another model.

The BSs have a distinct boundary of range. All jobs within this range are served regardless of their distance to the BS. Furthermore, we assume the signaling traffic between BSs and jobs to be reliable within the range of the BS.

Queues store the jobs for each BS. We neither limit the number of queues per BS, nor the number of connected BS per queue. We only limit the number of jobs per queue.

We do not consider limits on the communication between BSs which, therefore, know each others states at all times.

2) Jobs: The smallest unit of work considered in this paper is a user request called job. A job represents a complete downlink transmission, which can, for example, be a video stream, an e-Mail, or a file download. Each job is, thus, not only a single network packet but a sequence of packets. To be able to model this as a Markov chain, we assume that the service time of each job, as well as, the inter-arrival time of jobs is exponentially distributed. This might not hold in realistic scenarios as effects like the distance between jobs and BS, and the jobs’ size interfere with this assumption.

Furthermore, the size of an area determines the rate of arrivals in this area. For our model these assumptions are equivalent to the assumption that the arrivals form a space-time Poisson process.

Each job is uniquely assigned to one queue, according to its arrival position. We distinguish between refused and servable jobs. Jobs are refused when they arrive while their corresponding queue is full. The jobs will not be enqueued or served. All other jobs are servable.

3) Cell Selection: The BSs’ ability to communicate with each other to decide which one serves a job that arrived in an area that can be served by more than one BS is called Cell Selection (Area 2 in Figure 1(a)). Only one of the connected BSs serves the job. Cell Selection can be implemented by techniques like Dynamic Cell Selection [1].

4) Cooperation: Two or more BSs are able to increase their range by cooperating (dotted line in Figure 1(a)). Jobs in the intersection of two or more enhanced ranges can be served by cooperation (Area 4 in Figure 1(a)). When all involved BSs are finished working on a job in the cooperative queue, the job is served. In our model, it is not necessary that all BSs work on a job simultaneously.

In contrast to Cell Selection, using cooperation all cooperating BSs have to work on a job. Figure 1(b) shows this as a queuing system. Queue 4 can only be served by BSs A and B together.

B. Problem Statement

In this paper, we evaluate the efficiency of different strategies to coordinate BSs. The efficiency of each strategy is mea-
sured using the average latency \( W \) and compared with latency bounds. For reasons of simplicity we consider scenarios with two and three BSs with and without cooperation.

For each of the scenarios, we analyze the following processing strategies to serve the jobs: (1) Random, (2) Longest Connected Queue, and (3) Round-robin.

We also determine the trade-off between coverage and latency when using cooperation (as seen in Figure 1(a)) to check, whether the increased coverage justifies the increased latency.

IV. CONVERSION TO MARKOV CHAIN

This chapter explains each of the strategies as well as their modeling as a Markov chain. We also describe a method to model cooperation.

A. Strategies

Let \( Q = \{Q_1, \ldots, Q_q\} \) describe the set of queues in the model, where \( |Q_i| \) is the length of Queue \( i \). Each queue collects all arrivals in one area. As we are limiting the size of each queue to \( k \), it holds that

\[
|Q_i| \leq k, \forall i \in [1, q].
\]

Furthermore, let \( S = \{S_1, \ldots, S_s\} \) be the set of BSs.

The function which determines the affinity of queues to BSs

\[
f : Q \rightarrow \mathcal{P}(S)
\]

is given by the geometrical positions of the BSs. Similarly, \( g : S \rightarrow \mathcal{P}(Q) \) defines all connected queues for each BS.

To determine the percentage of time BS \( j \) works on Queue \( i \) each model defines a function \( v : (Q \times S) \rightarrow \mathbb{R} \). \( v \) depends on the state of the Markov chain, as the strategies may behave differently in each state. One can prove that using percentages directly at transitions, as we use it here, replaces using decision states.

Let \( \mu_j \) describe the service rate of BS \( j \). We compute the service rate for Queue \( i \) with \( |Q_i| > 0 \) as:

\[
r(Q_i) = \sum_{j \in f(Q_i)} v(Q_i, j)\mu_j.
\]

If more BSs want to work on a queue than jobs are in that queue, the percentages must be adjusted to ensure that each job is processed only once. To ensure that \( v \) is maximal, remaining percentage of BSs’ time is distributed to the next best queues.

Arrival rates for all Queues \( i \) with \( |Q_i| < k \) are defined as \( \lambda_i \).

1) Random: BSs implementing the Random strategy choose a job randomly after each job completion. The probabilities to select a given queue are fixed, but chosen depend on the arrival rate of each queue: Let \( \beta_{s,i} \) denote the probability that BS \( s \) chooses to work on Queue \( i \). Hence, \( \forall s \in S : \sum_{i \in g(s)} \beta_{s,i} = 1 \).

The probability to serve empty queues is set to 0, which increases the probabilities of serving the other queues in this case.

We model Random as a \( q \)-dimensional Markov chain. Each State \( s \) is defined as a tuple of queue lengths:

\[
s \in |Q_1| \times \cdots \times |Q_q|.
\]

The rate \( v(Q_i, S_j) \) for the transition \((|Q_1|, \ldots, |Q_i|, \ldots, |Q_q|) \rightarrow (|Q_1|, \ldots, |Q_i| - 1, \ldots, |Q_q|)\), \( \forall i : |Q_i| > 0 \) can be computed using

\[
v_{\text{Rand}}(Q_i, S_j) = \begin{cases} 0, & |Q_i| = 0 \\ \beta_{j,i} + \frac{\sum_{m \in g(S_j), |Q_m| > 0} \beta_{j,m}}{|Q_i| - 1}, & \text{else}. \end{cases}
\]

2) Longest Connected Queue: When using LCQ every BS prefers jobs in the longest queue connected to it. In case of equality of the longest queues the BS serves the queue that can be served by the least number of BSs, or randomly chooses one of those as last tiebreaker. Areas that can be served by less BSs are preferred, as there is less flexibility in serving them.

Analog to Random, the Markov chain of LCQ is \( q \)-dimensional with transitions:

\[
v_{\text{LCQ}}(Q_i, S_j) = \begin{cases} 0, & \exists Q_m \in g(S_j) : |Q_m| > |Q_i| \\ 0, & \exists Q_m \in g(S_j) : (|Q_m| = |Q_i| \wedge |f(Q_m)| < |f(Q_i)|) \\ \frac{1}{b}, & \text{else}; \end{cases}
\]

with \( b = \{q \in g(S_j) \wedge |q| = |Q_i| \wedge |f(q)| = |f(Q_i)|\} \).

A graphical representation of the LCQ Markov chain is given in Figure 2. The figure shows the service rates for a two-BS Markov chain.
3) Round-robin: Round-robin BSs alternate between all connected queues. To increase the performance the BSs switch the queue if their current queue is empty and there are other queues that can be served. The BS continues its round from the queue it jumped to.

A state in our Round-robin model shows the queue sizes as well as the next queue to serve for each BS. The function \( n : S \rightarrow Q \) determines the next queue to serve for each BS. It holds that \( \forall i \in \{1, \ldots, q\} : n(S_i) \in g(S_i) \).

To describe a State \( s \) the following tuple is used:

\[
s \in (|Q_1| \times \cdots \times |Q_q|) \times (n(S_1) \times \cdots \times n(S_q)).
\]  

(6)

The value of each transition

\[
(|Q_1|, \ldots, |Q_q|), (n(S_1), \ldots, n(S_q)) \rightarrow (|Q_1|, \ldots, |Q_q| - 1, \ldots, |Q_q|),
\]

\[
(n(S_1), \ldots, n(S_q) + 1 \mod |g(S_j)|, \ldots, n(S_q)),
\]

\( \forall i : |Q_i| > 0 \land \forall j : n(S_j) = Q_i \) can be computed using

\[
v_{RR}(Q_i, S_j) = \begin{cases}
1, & |Q_i| > 0 \land \forall 0 \leq v < t \land (n(S_j) + t \mod |g(S_j)|) = Q_i \Rightarrow (n(S_j) + v \mod |g(S_j)|) = 0 \\
0, & \text{else}.
\end{cases}
\]  

(7)

4) Extension for cooperation: To extend the models, described above, by cooperation, each Queue \( R \) for an area that uses cooperation is divided into \( |f(R)| \) sub-queues, one for each BS that has to cooperate to serve jobs from it. Figure 3 illustrates the idea.

Let \( R_i \) denote the exclusive sub-queue for BS \( i \) and \( |R_i| \) its length. A copy of each incoming job for \( R \) is added to every sub-queue. Each of the sub-queues is served in FIFO order. Therefore, two sub-queues of the same length contain the same jobs.

The number of remaining jobs in Queue \( R \) is the maximum length of all its sub-queues \( |R_i| \). For all computations with the Markov chains, the lengths of the sub-queues need to be ignored, as only the length of \( R \) is important.

With a rate of \( \lambda_R \) one job is added to each sub-queue, if none of them if full. Otherwise, the job is refused. Each of the \( R_i \) is served by exactly one BS, according to one of the strategies above. The priority of these queues is equal to a queue that can be served by \( |f(R)| \) BSs (needed as tiebreaker for LCQ).

This extension allows to model cooperative queues, that need to be served by more than one BS, as a set of normal queues that are only served by one BS. Therefore, nothing needs to be changed at the modeling of the strategies described above, in order to be able to numerically analyze cooperation.

B. Determining average latency

To derive the average latency \( W \), we calculate the stationary distribution as described by Gross [2] and Kleinrock [3] and compute the average number of jobs from the resulting distribution. From the average number of jobs in the system we determine the average latency using Little’s law [2], [3].

C. Latency bounds

Additionally to the selection strategies, we provide a lower bound for the latency by allowing the BSs to select jobs from the whole area (M/M/2 for two BS respectively M/M/3 in the three BS case). These systems share one queue with several BSs while our systems only share some of their queues. The BSs in this lower bound comparison will be fully utilized as long as there is work to do. A wrong decision of one of the strategies could potentially lower the utilization, leading to a higher latency.

D. BS Placement

BSs need to be placed in a hexagonal pattern to optimally cover an infinite plane with their circular coverage, leading to an optimal distance of \( \sqrt{3}r \) between the BSs [8], [9]. When using cooperation the BS coverage radius is increased by a constant factor depending on the path loss. In an earlier paper [4] we showed that a cooperation radius of \( \sqrt{2}r \) is possible with a path loss exponent of 2 leading to an optimal distance of \( \approx 2.19r \).

Using basic geometry, we calculate that the overlapping area is \( \approx 2.97\% \) in the two-BSs case and \( \approx 6.12\% \) of the total area in the three-BSs case with the optimal distance between the BSs.

V. Results

To determine the best strategy we compare the average latency of Round-robin, LCQ and Random in the two- and three-BSs scenario.

Using these results we quantify the trade-off between latency and coverage using the strategy with the lowest latency. Additionally, we quantify the impact of cooperation to latency and coverage. A comparison between the scenarios with and without cooperation is done.

A. Latency comparison

The limitation of the queue sizes during the numerical analysis makes it necessary to determine the probability to refuse jobs. By limiting further analysis to workloads that lead to less than 1% refused jobs we ensure realistic results. The maximum queue length of the M/M/2 (M/M/3) systems are set to the sum of all limits of the queues.

Figure 4(a) shows the number of refused jobs, with a maximal queue length of \( k = 30 \) in a two-BSs system. The BSs are overlapping in 2.97% of the total area as this is the optimal BS placement in terms of coverage. The number of refused jobs, using the random strategy, grows to \( \approx 3.62\% \) with a utilization of 100%. Therefore, we are limiting the
maximal workload to 90% to compute the average latency as shown in Figure 4(b).

We made the same computations for three BS with 6.12% Cell Selection area and \( k = 5 \). Due to the smaller queue size, 1% of the jobs are refused with at a 48% utilization as shown in Figure 5(a). The average latency for the three-BS case is shown in Figure 5(b).

LCQ has the best performance, and leads to at most 1.86 times higher latency than the lower bound in the two BS case (1.44 when considering three BS). The gap between LCQ and Round-robin/Random gets bigger with growing workload due to the fact that LCQ automatically adjusts to the different loads at each queue while Random and Round-robin statically serve each queue with a given probability.

We select LCQ as the best strategy for the trade-off analysis between covered area and latency.

**B. Trade-off between latency and coverage**

By varying the distance between BSs, we increase the area of Cell Selection which decreases latency, but also reduces coverage. Cooperation can be used to compensate for the loss of coverage when placing BS differently.

Cooperation increases the BSs’ range (dotted line in Figure 1(a)) while only the intersection of this range can be served (dotted and solid gray area in Figure 1(a)). Only the newly covered area (solid gray in Figure 1(a)) will be served using cooperation as the area which is covered by the conventional BS coverage can be served from a single BS already.

Figure 6 shows the covered area with and without cooperation for two BSs. The point of optimal BS coverage of an infinite plane (hexagonal pattern with distance \( \sqrt{3}r \)) is marked in the plot. We assume cooperation to increase the coverage radius by a factor of \( \sqrt{2} \).

Figure 7 shows the latency for LCQ with and without cooperation as function of the covered area for two BSs (load is normalized to 50% for an area of \( 2\pi \)). As the covered area cannot be mapped uniquely to one situation when using cooperation (Figure 6 shows different possible distances to cover given areas) we focus on the distances \( 0.5 \leq d \leq 2.237 \) as this covers the realistic scenarios as well as the coverage-optimal distance 2.189.

Increasing the covered area leads to a higher load that increases the latency monotonically in both scenarios (i.e. with and without cooperation).
As long as the area can be covered without cooperation, the latency is smaller when not using cooperation. This is due to the fact that jobs in the cooperation area need to be processed twice, which increases the load on the BSs, leading to a higher average latency.

It can be seen that it is not useful, in terms of latency, to use cooperation when placing BSs densely. On the other hand, cooperation can be used to thin out the number of active base stations by ≈ 37.4% [4] for covering an infinite plane. The cost of this increased coverage is an increased latency. The latency in this two-BSs scenario using cooperation is ≈ 18.86% higher than without cooperation.

As the use cases differ (different requirements on latency, different load) there is no generally optimal way on placing BSs and whether to use cooperation or not, but we provided the means to quantify the trade-off between latency and number of BSs in this paper.

VI. CONCLUSION

Our results show that placing base stations (BSs) with highly overlapping areas and, therefore, a high potential for Cell Selection leads to a low latency for all considered strategies. The downside of increasing the amount of Cell Selection is the smaller total covered area, i.e. needing more BSs to cover the same area. Considering this trade-off we evaluated the influence of cooperation to coverage and latency. The number of active BSs can be decreased, without loosing coverage, on the cost of higher latency.

We showed that Longest Connected Queue (LCQ) has the lowest latency in a dynamic scenarios with exponential distributed inter-arrival and service times in comparison to Random and Round-robin. They have higher average latencies due to the lack of adaption to the current queue fill states. LCQ is the best of the evaluated strategies and should be used to efficiently serve multiple queues per BS, when using cooperative BSs.

The trade-off between latency and number of BSs needed to cover a certain area must be made when planning a radio access network. Cooperation allows to extend this trade-off to even less BSs, if higher latencies can be accepted. The models provided in this paper allow comparing the impact of Cell Selection and cooperation on the latency.

REFERENCES