Received: 21 March 2017 | Revised: 5 September 2017 | Accepted: 10 October 2017

DOI: 10.1002/nem.2014

RESEARCH ARTICLE

WILEY

Distributed dynamic load balancing with applications in radio access networks

Per Kreuger | Rebecca Steinert | Olof Görnerup | Daniel Gillblad

Swedish Institute of Computer Science (Rise SICS), Kista, Sweden

Correspondence
Per Kreuger, Rise SICS, Box 1263, SE-164 29 Kista, Sweden.
Email: piak@sics.se

Funding information
Swedish Foundation for Strategic Research, Grant/Award Number: RIT15-0075; Commission of the European Union, Grant/Award Number: 671639

Summary

Managing and balancing load in distributed systems remains a challenging problem in resource management, especially in networked systems where scalability concerns favour distributed and dynamic approaches. Distributed methods can also integrate well with centralised control paradigms if they provide high-level usage statistics and control interfaces for supporting and deploying centralised policy decisions. We present a general method to compute target values for an arbitrary metric on the local system state and show that autonomous rebalancing actions based on the target values can be used to reliably and robustly improve the balance for metrics based on probabilistic risk estimates. To balance the trade-off between balancing efficiency and cost, we introduce 2 methods of deriving rebalancing actuations from the computed targets that depend on parameters that directly affects the trade-off. This enables policy level control of the distributed mechanism based on collected metric statistics from network elements. Evaluation results based on cellular radio access network simulations indicate that load balancing based on probabilistic overload risk metrics provides more robust balancing solutions with fewer handovers compared to a baseline setting based on average load.

1 | INTRODUCTION

Distributed algorithms and self-organisation have been successfully used for many network management tasks, eg, routing, service discovery, and failure recovery, and often exhibit clear scalability and reliability advantages. However, for effective control and coordination of infrastructure resources, the trend in modern network management tends more towards (logically) centralised solutions implemented in programmable networking environments. Although a centralised management paradigm better supports optimised resource management and service deployment strategies, real-time systems and services still depend on highly distributed management functions for scalable and timely networking operations. One fundamental design challenge in future management of networked systems involves finding a balance between distributed and centralised network operations, another identifying high-level abstractions for controlling and representing the state of heterogeneous infrastructures.

We propose a generic approach to dynamic balancing of resources in networked systems that addresses these challenges based on the notion of distributed target computation (DTC)[1,2] operating on probabilistic risk metrics. We introduce risk as a metric for representing the probability of over-consuming a user-specified fraction of a given resource. Risks are examples of high-level abstractions needed for centralised decisions, while parameters in the metrics and rebalancing tactics open up for centralised control of essentially distributed mechanisms. In a network setting, metrics and targets are

computed locally at individual network entities (or nodes) using information from neighbouring entities. In the case of load balancing, a probabilistic risk metric relates to the risk of overloading a serving network entity (eg, an access point [AP]).

The main benefits of DTC include simplicity, scalability, and robustness. Distributed target computation is naturally distributed in the sense that the metrics and targets are computed locally by, or for, each single node using only information obtained from other nodes in its neighbourhood. The benefit of balancing of a probabilistic metric entails a unified representation and an abstraction of the network state and capacity, in contrast to dealing with vendor-specific metrics that are often varying in range and are difficult to compare. Additionally, probabilistic risk estimates account for variability in the observations, which leads to a more robust actuator and balancing mechanism.

Although we focus mainly on load balancing in the context of radio access networks (RANs), the approach is applicable to a range of resource management applications (eg, in cloud systems and traffic management) and resource metrics (eg, compute power, buffer memory, and/or node connectivity).

1.1 | Contribution

In Kreuger et al,[1] we presented a basic DTC in the context of LTE using one particular target load metric and a specific way of producing the cell range expansion (CRE) bias values. That method proved to be very effective, but relatively insensitive to cost incurred by the CRE changes. In this paper, we address this issue by using probabilistic risk of high-load levels as an alternative metric and a generalised interface for controlling rebalancing actions, based on the expanded exposition of the principles of the balancing approach.

The core contribution is the generic algorithm and modelling principles of the distributed balancing approach. The applicability of the approach is evaluated in a cellular LTE setting and further exemplified for WiFi systems. The generality and uniformity of the proposed metrics make them ideal also for centralised management in virtualised controllers in a multi-RAT (Radio Access Technology) scenario.

More specifically, the main contributions of this paper are as follows:

- A general method for dynamically balancing a metric between nodes in a distributed system.
- Application of the method to balance radio resource load in RANs using probabilistic risk estimates and a remotely configurable rebalancing mechanism.

After a short overview of relevant state of the art, we will present DTC in full generality, then specialise and evaluate it for the RAN load balancing problem. This is followed by a discussion and conclusions.

2 | RELATED WORK

Load balancing is used in several areas outside networking, eg, to plan resource allocation in data centres, work schedules, and industrial production, as well as some in networking not touched upon here, eg, routing. In some of these cases, centralised decisions[3,4] and static methods[5,6] are feasible. For others, including many in networking, dynamic methods[4,7] and distributed decisions[8,9] are more suitable.

Mobility balancing for cellular networks is a well-studied topic. Early proposals for cellular RAN include Chen,[10] which used adaptive cell sizing through cell transmit power regulation (physical power "breathing") to offload user equipments (UEs) from highly loaded cells, and similar approaches have more recently been proposed[11] for WiFi. The CRE mechanisms in LTE allow a similar rebalancing actuations, but without the complexity of physical power management.

Another distributed approach is described by Du et al,[12] where it is used for negotiating area coverage between base station agents. It is based on bilateral exchanges between base stations but uses complex cost/benefit estimates for each of a large number subareas coverable by several cells. Base stations negotiate exchanges of subareas to maximise benefit over cost in a 2-step multiagent process and synchronously commit to the most favourable ones. By contrast, our approach never modifies physical coverage, but only the CRE bias parameter, and exchanges between nodes that implement a rapid gradient descent involving *all* neighbours of a single node instead of a multistage agent bidding process.

Several authors propose physically centralised solutions, eg, Lobinger et al[13] who report work where load balancing is performed by considering individual user "satisfaction" based on both signal to interference and noise ratio and node radio load. They propose a method based on selecting individual candidate UEs and target cells for offloading, evaluating the effect on total "satisfaction," before committing to each handover. Another centralised approach[14] by Siomina and

Yuan introduces a method based on integer programming, to assign CRE values to each node, given load levels of the entire network. The method requires collecting and transferring load estimates to a central location, where a potentially time-consuming optimisation mechanism can determine suitable values for the CRE parameters. It is unclear how the authors intend to handle the delays and scalability issues implied by such a method. Similar issues arise in the centralised approach described in Wang et al,[15] which uses enforced handovers rather than manipulation of the CRE parameter.

Hao et al[16] propose managing individual handovers that has the potential to improve balance more efficiently than physical breathing, CRE-based methods, or indeed any global method treating UEs as a collective. As in the case of Lobinger et al,[13] this comes at a considerable cost in terms of management overhead and computational complexity.

Our approach is based on the methods presented in Kreuger et al[1] but introduces balancing overload risk and an actuation mechanism based on a parametrised sigmoid mapping. These are useful as a state abstractions and as controller APIs for logically centralised control within a unified management paradigm for heterogeneous infrastructures.

From a programmability perspective, several approaches and architectures for flexible management of software-defined RANs have been proposed.[17-19] The proposed architectures are aimed to enable logically centralised control for dynamic resource management and RAN slicing.[17] Moreover, this paradigm allows for effective coordination of radio resources and interference management. A logically centralised controller plane may consist of various types of controllers capable of operating at different timescales to perform control operations both in real time and at longer timescales.[17,19] However, scalability of centralised approaches remains an issue.

The concept of programmability abstractions[20] (here, realised as load metrics) is central for effective coordination and control of heterogeneous radio infrastructures and goes hand in hand with the development of architectures for software-defined RANs. The probabilistic load metrics proposed in our work are instances of such abstractions. The combination of logically centralised control and abstract representation of the network state enables automated and dynamic resource allocation fulfilling specified service and user requirements. Since load balancing in general is fundamental to ensure bandwidth and delay requirements, abstractions representing risk of overload are highly relevant to proactively avoid service unavailability and performance degradations.

3 | METHOD

We represent a dynamic system as a graph (Figure 1) where nodes constitute resources with an associated usage or load state, encoded as a single value $0 \leq m_i \leq 1$, its *metric*, and edges represent neighbourhood relations. In the RAN case, neighbourhoods correspond to cells with overlapping coverage, but although the complexity of the computation depends on the connectivity of the graph, in principle, any pairwise node relation is admissable.

How the usage state is generated does not concern us at this point, but we assume we can influence it through an actuation method parametrised by local bias values derived as the difference between the usage metric and a corresponding *target* value t_i computed for each node by DTC.

We first present the method in a form where the balancing metric m_i is left as a parameter and assume that the rebalancing mechanism takes a single actuation bias value per node as input, but, for generality, use a function $\mathcal{B}$, to derive that value from the node state. For any concrete balancing problem, m_i and $\mathcal{B}$ must be defined separately, and we will do so in full detail for the RAN load balancing scenario in Section 4.

In summary, we presume the following:

3.1 | Prerequisites

1. A way for each node to identify a *neighbourhood* of other nodes.
2. A current *metric* value m_i for each node i.
3. A mechanism for nodes to *exchange* target and metric values bilaterally.
4. A mechanism to influence the choice between nodes to serve client demands.

3.2 | Local neighbourhoods

We maintain for each node a *local neighbourhood*, ie, a list of related nodes with which it exchanges target and metric values. The neighbourhood can be either preconfigured or built up and updated dynamically based on observed node interactions, eg (following a scheme introduced in Kreuger et al[21,22]:

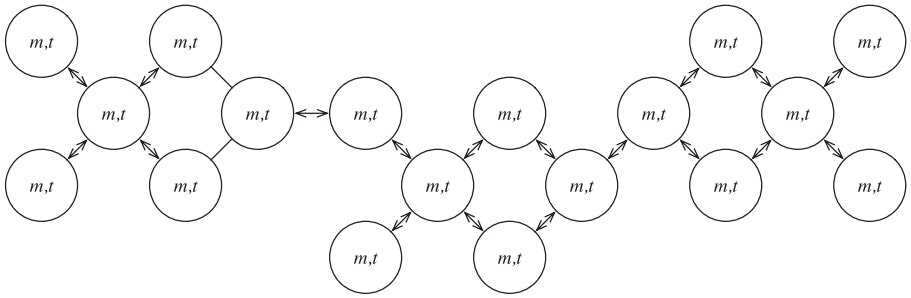


FIGURE 1 Example system graph. The system state is represented by the metric and target values associated with the nodes of the graph. State transitions are either metric updates or recomputation of the distributed target computation targets

Model the probability p_{ij} of each node i having an interaction with another node j by an estimate $\hat{p}_{ij}$. Assign for each node j in a set $j \in J_i$ of nodes with which i can be expected to have interactions* a prior

$$\hat{p}'_{ij} = \frac{1}{|J_i|}, \tag{1}$$

and initialise interaction counters $c_{ij} = 0$ and $c_i = 0$.

Increment for each observed interaction between i and j and counters c_{ij} and c_i, and estimate the probability of an interaction between i and j as follows:

$$\hat{p}_{ij} = \frac{w\hat{p}'_{ij} + c_{ij}}{w + c_i}, \tag{2}$$

where w is a weight controlling the decay rate of the estimate. Here, $\hat{p}_{ij}$ represents the expectation of the posterior distribution assuming a Dirichlet prior with parameters wp_{ij} and observation counts c_{ij}. In addition, replace all priors by the estimated posteriors, $\hat{p}'_{ij} = \hat{p}_{ij}$, and reset counters c_i and all c_{ij} after a fixed number of observations. This gives a smooth exponential decay of the influence of older observations while ensuring that the estimates for all $j \in J_i$ sum to 1.

Finally, let the *local neighbourhood* $\mathcal{N}_i$ be the *smallest* subset of J_i such that

$$\sum_{j \in \mathcal{N}_i} \hat{p}_{ij} > n, \tag{3}$$

for some minimum value n on the joint probability of an interaction with a node in $\mathcal{N}_i$, *and* so that, for any $l \in J_i \backslash \mathcal{N}_i$, for all $j \in \mathcal{N}_i$, $\hat{p}_{il} \leq \hat{p}_{ij}$. $\mathcal{N}_i$ will thus, according to the estimates, consist of the most likely nodes $j \in J_i$ for i to have interactions with.

3.3 | Metrics

The type of momentary measurement statistics used to derive the metrics will differ with the application, but generally, we expect the metrics to be probabilistic estimates produced from periodic observations of some variable closely related to the balancing objective. We assume that it is updated regularly and is available to the DTC update mechanism at any time.

We will see several examples of concrete metrics in Section 4.1.1, but for this formulation of DTC, the details of how the m_i are produced or collected are not essential.

3.4 | Distributed target computation

We update the target value for each node triggered by the detection of some event at the node. The event can be either a timeout, a significant change in the metric value, an interaction with another node, or requested by an external mechanism. The computation of target values is distributed between the nodes in the network, but any individual node interacts only with the nodes in its neighbourhood. Computation of target values can be done by either polling or pushing target

*It is not crucial that J_i initially includes *all* possible nodes, since any one $l \notin J_i$ can be added as needed by creating a new counter c_{il} and assigning (at the cost of a negative initial bias) to it a prior $\hat{p}'_{il} = 0$.

values between neighbours. We will describe an *on-demand polling* version of the computation, but it is not difficult to imagine an asynchronous push variation, which may be preferable for some applications.

3.4.1 | Local target update

Whenever a node i needs to update its current target t_i, it executes the following procedure:

1. Retrieve, for each node $j \in \mathcal{N}_i$, their current *target* values t_j.
2. Adjust i's *target* value t_i to the *mean* of the local metric value m_i and the retrieved target values t_j:

$$t_i^n = \frac{m_i + \sum_{j\in\mathcal{N}_i} t_j^{n-1}}{1 + \left|\mathcal{N}_i\right|}. \tag{4}$$

3. Request that all j in i's neighbourhood $\mathcal{N}_i$ adjust their targets t_j as in step 2.
4. If t_i differs from its previous value by less than a cut-off value, terminate the procedure.
5. If not, and unless a maximum number of iterations have been reached, go to step 1.

3.4.2 | Convergence

The update operation for a single node i iterates over i and $\mathcal{N}_i$, interleaving adjustments of their target values t_i and t_j for $j \in \mathcal{N}_j$. Provided that the metrics m_i and all m_j for $j \in \mathcal{N}_i$ remain fixed, and that updates of the t_j consider only target changes within $\mathcal{N}_i$, each involved node adjusts at each step its target to reduce the value of a convergence metric:

$$\left| t_i - \frac{m_i + \sum_{j\in\mathcal{N}_i} \left| t_j - \frac{m_j + \sum_{k\in\mathcal{N}_j} t_k}{1+\left|\mathcal{N}_j\right|} \right|}{1 + \left|\mathcal{N}_i\right|} \right|. \tag{5}$$

Each application of Equation 4 to a node i or one of the $j \in \mathcal{N}_i$ eliminates its direct contribution to the convergence metric of i but may, due to the node's occurrence in neighbourhoods of other nodes within $\mathcal{N}_i$, increase those of other nodes. Repeated interleaved application will however eventually reduce the metric to zero. This is true for the node i itself but not necessarily for all nodes in $j \in \mathcal{N}_i$ since their convergence metric may also depend on nodes outside $\mathcal{N}_i$.

The number of steps required for the convergence of a single node depends on the connectivity within its neighbourhood, but for RAN load balancing and typical HetNet scenarios, where only macro nodes have handovers with more than a few other nodes, the total number of iterations needed to reduce the convergence metric to less than 10^{-3} is rarely more than ~10. Within each iteration, the number of messages is bounded by $\left|\mathcal{N}_i\right|^2$ but is typically closer to $\left|\mathcal{N}_i\right| \log(\left|\mathcal{N}_i\right|)$.

If all nodes are updated regularly, and local metrics remain fixed, the entire system will eventually converge towards a global equilibrium where Equation 5 evaluates to zero for every node k, and the sum of all target to metric offsets $\sum(t_k - m_k)$ is also zero. For a large, highly connected system, complete convergence may take considerable time, depending on the initial distance from the equilibrium state. On the other hand, the local improvement is immediate and often considerable, and in a dynamic system, where metrics change continuously, the system will always tend towards a target distribution that approximates the global equilibrium. Typical performance of systems of up to 15 nodes are presented in Section 5.

The following section presents the update mechanism in more detail.

3.4.3 | Algorithm

One step target adjustment

Steps 1 to 2 of Section 3.4.1 are detailed in Algorithm 1: On receiving the message "adjustTarget(*neighbourhood*)", a node requests updated target values from its neighbours, then updates its local target value.

Algorithm 1 The "adjustTarget" and "getTarget" procedures

```
Require: myNeighbours, myMetric
  on receipt of: adjustTarget() do
    sum ← myMetric
    n ← 1
    for all node ∈ myNeighbours do
      trigger getTarget(self) @ node
      on receipt of: target(node, nTarget) do
        sum ← sum + nTarget] ; n ← n + 1
      end do
    end for
    myTarget ← sum/n
  end do
Require: myTarget
  on receipt of: getTarget(node) do
    trigger target(self, myTarget) @ node
  end do
```

Neighbourhood-wide target update

The complete local target update routine, including the iteration of steps 3 to 5, is illustrated in Algorithm 2: On receiving message "updateTarget()", a node first adjusts its own target value as in Algorithm 2, then request the nodes of its neighbourhood to do the same, but using updated targets only from within *myNeighbours*. If the resulting target value differs by more than *cutoff* from the previous one, the procedure is repeated.

Algorithm 2 The "updateTarget" procedure

```
Require: myNeighbours, myMetric, cutoff
  on receipt of: updateTarget() do
    repeat
      old ← myTarget
      trigger adjustTarget() @ self
      for all node ∈ myNeighbours do
        trigger adjustTarget() @ node
      end for
    until |old − myTarget| < cutoff
  end do
```

3.5 | Biasing

How we actuate a change in the balance of a chosen metric depends strongly on the application domain, but since we here consider primarily systems where distributed nodes serve client demands, we restrict the presentation to mechanisms that depend on a single actuation bias to skew the choice between nodes to serve clients and refer to a function mapping the node state to such a value as a *biasing function*.

In principle, any aspect of the state of a node and its neighbourhood can be used as input to the biasing function, but since DTC produces a local target to metric offset for each node, we will focus on biasing functions that takes these as inputs. The simplest biasing function just returns the local offset, but as we will see in Section 4, scaling the local offset and using the distribution of offsets within the neighbourhood can improve the actuation performance and reduce rebalancing cost.

Algorithm 3 is thus parametrised by a biasing function $\mathcal{B}$, a set of *control parameters* and the local and neighbourhood offsets produced by Algorithms 1 to 2.

Algorithm 3 The "updateBias" and "getMetric" procedures

Require: $\mathcal{B}$, *myNeighbours*, *myMetric*, *myTarget*, *control parameters*
  on receipt of: updateBias() **do**
    for all *node* ∈ *myNeighbours* **do**
      trigger getOffset(*self*) @ *node*
      on receipt of: offset(*node*, *nOffset*) **do**
        offsets $\left[node\right]$ ← *nOffset*
      end do
    end for
    actuationBias ← $\mathcal{B}$ (*myTarget* − *myMetric*, *offsets*, *control parameters*)
  end do
Require: *myMetric*, *myTarget*
  on receipt of: getOffset(*node*) **do**
    trigger offset(*self*, *myTarget* − *myMetric*) @ *node*
  end do

Algorithm 4 is an example code entry point for a node, assuming actuation reads and uses the bias produced by Algorithm 3. It initialises and updates the local target and bias and then goes to sleep, awaiting the next update trigger.

Algorithm 4 Node initialisation and example update trigger

Require: *myNeighbours*, *myMetric*
  myTarget ← *myMetric*
  trigger updateTarget() @ *self*
  trigger updateBias() @ *self*
  whenever: a significant change in *myMetric* is detected, **do**
    trigger updateTarget(*cutoff*) @ *self*
    trigger updateBias() @ *self*
  end do

4 | RAN LOAD BALANCING

We will examine 2 network load balancing applications of which the cellular case (LTE) is by far the most detailed. The WiFi case is covered by an outline of one way, a corresponding mechanism may be implemented.

4.1 | LTE load balancing

In LTE, the requirements of Section 3.1 can be fulfilled by fairly straightforward extensions of existing mechanisms. Requirement 1 can be covered by simply preconfiguring the neighbourhood of each node at deployment time or by implementing the method described in Section 3.2, based on simple handover counters for pairs of nodes. Requirement 3 needs a simple node-to-node communication protocol (eg, an extension to the X2 protocol[23]) to allow requests for and reports of 2 high precision numbers, or a centralised mechanism that can emulate such exchanges for the nodes in a particular region. Once that is in place, the distributed computation of target values is straightforward to implement, more or less directly as stated in Algorithms 1 to 2. Requirement 4 is fulfilled by the existing CRE parameters of LTE (and CDMA [code-division multiple access]) and the standard handover mechanism, although several other biasing mechanisms may also be envisioned. Section 4.1.2 will discuss two based on dynamic CRE adjustments, but first, we will examine some of the details of how to fulfil Requirement 2.

4.1.1 | Load metrics

In network load balancing, we will always start out with measurements of some type of resource usage. The resource could be buffer memory, decoding, forwarding, or compute resources, or in the case of RANs, the radio bandwidth resource.

Here, we will only consider the radio resource, but other and combined measures are certainly also possible. In the LTE case, we will assume that we have access to momentary radio resource load measurements produced at a regular intervals (eg, few times per second) in terms of the number of *used* resources blocks divided by the number of *available* resource blocks, ie, a number between 0 and 1. Since such measurements typically fluctuate very rapidly, and we expect to regulate load on timescales of several minutes, we will consider 2 types of probabilistic metrics based on statistics of the underlying measurements.

Load mean metric

Define an exponentially decaying moving average

$$\hat{l}_k = \frac{w\hat{l}_{k-1} + n\bar{l}_k}{w+n}, \tag{6}$$

where $\bar{l}_k = \frac{1}{n}\sum_{i=(k-1)n}^{kn-1} l_i$ is the sample mean of momentary loads l_i for the estimation period, ie, $(k-1)\,n \leq i < kn$, and $\hat{l}_0$ is a prior initialised to any number between 0 and 1. We update the prior expectation $\hat{l}_{k-1}$ after a fixed number of observations n, so that $k = \lfloor {}^{i}/_{n} \rfloor$. Keeping n (and/or w) relatively large will tend to smoothen variations of the metric and reduce handovers caused by rapid fluctuations of the momentary load. The advantage over a traditional exponentially weighted moving average, where $n = 1$, is that we have separate control over the size n of the window over which we estimate, and the decay rate ${}^{1}/_{w}$.

High load risk metric

As a refinement, we will introduce an estimate of the risk of high loads using a moment estimation scheme where the first load moment is estimated as in Equation 6 and the second moment (variance) by a corresponding exponentially decaying moving variance estimate. Assuming a Gaussian-Whishart prior,[24] the expectation of the posterior variance distribution and the corresponding exponentially moving variance estimate can be written as

$$\hat{v}_k = \frac{w\hat{v}_{k-1} + \sum_{i=(k-1)n}^{kn-1}\left(l_i - \bar{l}_k\right)^2 + \frac{wn}{w+n}\left(\hat{l}_{k-1} - \bar{l}_k\right)^2}{w+n}, \tag{7}$$

where $\bar{l}_k$ is again the sample mean of momentary loads, $\hat{v}_{k-1}$ and $\hat{v}_k$, the prior and posterior expectation of the variance, respectively, and $\hat{l}_{k-i}$, the prior expectation of the mean, as obtained in Equation 6 above. The subexpressions $\sum_{i=(k-1)n}^{kn-1}\left(l_i - \bar{l}_k\right)^2$ represent sample variance over the estimation period, and $\frac{wn}{w+n}\left(\hat{l}_{k-1} - s_k\right)^2$ a term compensating for the gradual shift and decay of the mean estimate.

Using these 2 moment estimates, we then estimate the parameters of a log-normal model of the load distribution of the node using the method of moments:

$$\begin{cases} \hat{\mu}_k = \ln \hat{l}_k - \frac{1}{2}\hat{\sigma}_k^2 \\ \hat{\sigma}_k^2 = \ln\left(1 + \frac{\hat{v}_k}{\hat{l}_k^2}\right) \end{cases}, \tag{8}$$

and use the cumulative distribution function of the estimated distribution to extract a momentary risk estimate $\hat{c}_k$ of observing loads over a given fraction φ (eg, 95%) of the node capacity r.

$$\hat{c}_k = P(L > \varphi r) = \frac{1}{2} - \frac{1}{2}\mathrm{erf}\left[\frac{\ln \varphi r - \hat{\mu}_k}{\sqrt{2}\hat{\sigma}_k}\right]. \tag{9}$$

A note on the choice of a log-normal model

Log-normal empirical distributions are common in the telecom domain. For example, link-level throughput measurement often fits well to log-normal distributions (see, eg, Fukuda,[25] Downey,[26] and Kreuger and Steinert[27]). Although there is

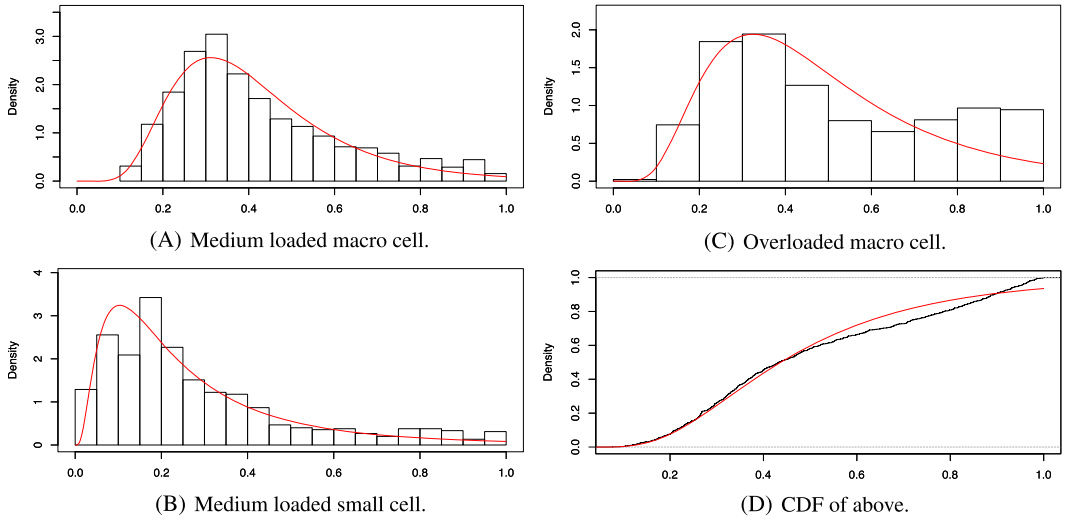


FIGURE 2 Load distribution with log-normal fits for individual cells (random 3-minute extracts from simulator runs). Note that although the empirical distribution in the right hand case is clearly bimodal, the value of the cumulative distribution function (CDF) of the fitted distribution in the region just below the cell capacity is in fact quite accurate

no a priori reason to assume that LTE radio cell load levels are also log-normal,[†] our simulator experiments appear to support this assumption—see Figure 2. The load generating process in the simulator is the combined effect of Pareto-fitted individual UE traffic variations and the UE mobility patterns. The momentary UE density distribution is a function of original UE "home" positions and Levý walk style (heavy-tailed flight distances[29]) movements. Even so, the resulting load distributions do conform well to log-normal distributions, at least up until the point where a significant part of the demand remains unfulfilled due to link-level congestion. This is a pattern that is familiar from studies of (fixed) link load measurements, where the distribution becomes bimodal at high loads and where the secondary mode appears close to, but below, the resource capacity limit. In this study,[27] we also found that the probability mass of the secondary peak is often approximated well by the mass of the tail of a fitted distribution that extends beyond the capacity of the resource, approximating the *demand* distribution.

4.1.2 | LTE biasing

LTE CRE parameter biasing by maximal scaling

To redistribute the load of the network towards the computed target t_i for each node i, one way is to assign CRE bias values to each node in a suitable range (eg, $[-3, 3]$ dB) in a way which uses as much as possible of an available CRE range locally, while preserving correct gradients between the nodes within each neighbourhood.

We propose to do so by using the following bias function

$$\mathcal{B}\left(b_i, \left\{b_j\right\}_{j\in\mathcal{N}_i}, range\right) = \frac{|range|\,(b_i - \epsilon)}{\hat{b} - \check{b} - 2\epsilon}, \tag{10}$$

using only the $\max \hat{b} = \max\{b_i\} \bigcup \left\{b_j\right\}_{j\in\mathcal{N}_i}$ and $\min \check{b} = \min\{b_i\} \bigcup \left\{b_j\right\}_{j\in\mathcal{N}_i}$ of the neighbourhood offset distribution.

The interval $\left[\check{b}, \hat{b}\right]$ represents the range of offsets that we need to represent within the neighbourhood, and we use that to scale the corresponding local offset b_i. This has the advantage of using the entire range of acceptable CRE values but tends to give large swings as the maximum difference in the neighbourhood approaches zero. We reduce this tendency by using a cut-off ϵ on $\hat{b} - \check{b}$, beyond which we avoid adjusting the CRE value.

As will be seen in Section 5, this biasing mechanism is very effective in terms of the balance achieved, but also costly in terms of handovers per connected user. Since it exploits the full range of CRE values, there is also a high risk of serving some users with less than the optimal link quality.

[†]Nor have we found any published analysis of cell level load distributions, either supporting or falsifying any such claim. Naboulsi's[28] otherwise excellent overview of mobile traffic analysis results and data sources has no mention of resource load distributions.

Ideally, we should actuate the rebalancing mechanism only when the gains to the overall performance of the system are significant and only when it can be done at a reasonable cost to individual UE's channel quality. Switching to balancing risk of high loads is one step in this direction. Another is to scale the offset b_i more smoothly, and so that only large differences in the metrics translate to large difference in CRE bias.

LTE CRE parameter biasing by sigmoid mapping

The cost of load balancing can be quite high in terms of handovers and potentially also in terms of reduced link quality for a UE that is forced to connect to a cell other than the one with the best signal. For this reason, it is desirable to reserve the highest CRE settings for situations where the load balance is very bad. The mechanism of Section 4.1.2 takes no account of this but simply tries to equalise the chosen metric by as large a range of CRE values as possible. An alternative is to scale the difference b_i by means of a function that smoothly tapers off as it approach the endpoints of the allowable range of CRE values. Sigmoid functions, such as the tangens hyperbolicus, turn out to be ideal for this purpose, especially if we augment them with a scaling constant that influences the steepness of the slope close to $b_i = 0$. The function we use for this purpose is defined as follows:

$$\mathcal{B}\left(b_i, \{b_j\}_{j\in\mathcal{N}_i}, m, k\right) = m\frac{\tanh kb_i}{\tanh k}, \tag{11}$$

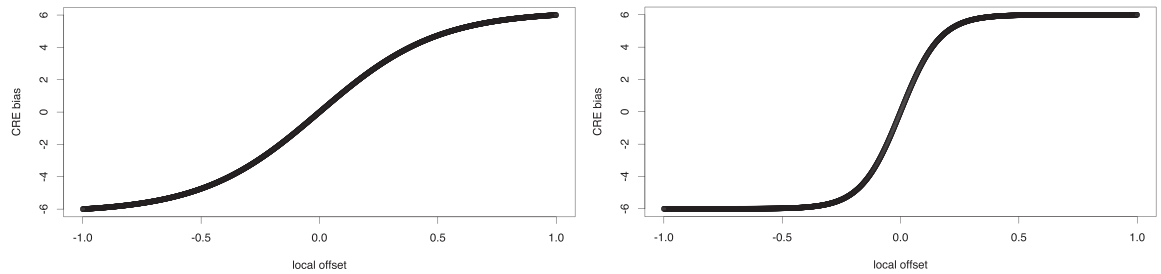
where k is the scaling constant mentioned and m is a constant used to scale the $[-1, 1]$ range of the tanh function to the allowable range of CRE bias values, ie, 3 for ±3 dB. Although, in principle, the mean metric in the neighbourhood of a highly loaded cell can be close to one, the mean target will include a component of the target load of the current node, so the differences close to the extreme values of the $[-1, 1]b_i$ value range are extremely rare. Figure 3 shows a plot for the function for $m = 6$, and $k = 2$ and $k = 6$.

Since we can arbitrarily choose how eagerly we scale the b_i value to CRE settings, we can have anything from a linear mapping of b_i values to CRE range (for $k \leqslant 1$) to a mapping that effectively reproduces the results of the mechanism of Section 4.1.2.

Most interestingly, perhaps, is that we can extend the allowable range of CRE values beyond the ones desirable in normal practice and reserve the extreme values for cases where the load situation is such that the server with the best channel quality is so overloaded that connecting to it will provide no practical link quality advantage. Exactly at which point this occurs in real networks needs careful analysis of recorded data and practical experiments, but given such knowledge, scaling with sigmoids is a promising candidate to implement balancing policies. We will explore a few examples from the parameter range of Equation 11 in Section 5, but because of the simplicity of our radio model, these should be regarded as indicative, rather than conclusive.

4.2 | WiFi adaptation

The load balancing method described in Section 4.1 was originally designed[1] for LTE, and even though it has not yet been evaluated outside that context, an analysis of the requirements of the method shows that it can in fact quite easily be adopted to other RATs. For example, in CDMA cellular networks, an almost identical method can be implemented with very similar effort, although exactly how the momentary measurements are obtained and the communication between



(A) $k = 2, m = 6$ (B) $k = 6, m = 6$

FIGURE 3 Examples of 2 parameter settings for the sigmoid mapping of offset to cell range expansion (CRE) range

the nodes is implemented may differ slightly. The same goes for any future cellular system that implements a CRE type of mechanism.

In WiFi, the IEEE 802.11 standard[30] and its amendments, including 802.11r, 802.11k, and 802.11v, support roaming within managed networks of APs served by one or more local controllers. These manage IP to MAC addresses translation, secure forwarding of connections, fast transition though predisconnection key exchange, radio state probing, and link state sharing between APs, and assisted roaming through AP neighbour reports for clients and explicit transfer requests from both clients and APs.

The neighbourhood reports are based on collected RSSI probe statistics and configured AP topology and can be specialised for each AP and client. The combination of client neighbourhood reports and explicit transition requests gives the network a lot of control of the client roaming behaviour, but there is nothing stopping the client from spontaneously issuing transition requests, eg, based on probes of the APs listed in the reports. The client roaming decision is not as standardised as in cellular, but even for clients without the assisted roaming extensions, transitions can be enforced by explicit disconnection and selective reconnection at more desirable APs.

This gives us several options for implementing our load balance proposal, but we will here only outline a version that is very similar to the one for LTE and indicate the extensions required over existing mechanisms.

1. The 802.11k mechanism for generating neighbourhood reports based on RSSI probe statistics is extended to also generate local neighbourhoods for the APs based on transition statistics, as in Section 3.2.
2. The facility for sharing link state information between APs in 802.11k is extended with the metric and target value exchanges used in the algorithm of Section 3.4. Alternatively, metric values, or the statistics needed to compute them, are collected at the controller and used by a centralised version of the algorithm. The metrics used can be the same as, or variants of, those proposed in Section 4.1.1.
3. Bias values are computed with one of the proposals in Section 4.1.2 and sent out to the clients via an extension of the 802.11k neighbourhood report.
4. The clients actuate the rebalancing operation by spontaneously sending a 802.11v transition request whenever its RSSI + bias becomes less than that of a neighbouring AP, according the report. Alternatively, the request can be sent from the controller or the AP, preceded by a neighbourhood report indicating the desired target AP(s).

Implementing DTC for WiFi requires similar implementation effort as for LTE, but extensions of the existing standards are more extensive and involve client changes.

Further steps towards coordinated resources in WiFi networks include programmability frameworks. Recent proposals are based on the concept of light virtual APs (LVAPs), abstracting the connection between the client and serving node, while simplifying state management following from client-AP (re)association (eg, previous studies[31-33]). The DTC approach could in this context, for example, serve the purpose of managing (by associated target values) the number of LVAP instances hosted per node, triggering instantiation or migration of LVAPs between serving nodes when needed. A full-fledged distributed implementation of DTC in a programmable framework would require local message exchanges in the control plane between neighbouring nodes (hosting control agents) for the purpose of updating target values, whereas a central control instance would set various parameters (eg, actuator thresholds) influencing the overall behaviour of DTC (see also Section 6.4).

5 | EXPERIMENTS

To evaluate DTC in the RAN load balancing scenario under diverse conditions in a controlled and reproducible manner, we have used simulations. We have examined the performance of several versions of the method in a wide range of scenarios. Each scenario repeats the user movement and traffic demand variations to compare versions of the method and emulates the connections and traffic load of a set of RAN nodes. The system model consists of the following components.

5.1 | Model system

5.1.1 | Traffic model

The traffic model is based on a collection of random processes generating (downlink) traffic bursts. The simulator maintains one such process per simulated UE, where the rate of the burst arrivals is sampled from a log-normal distribution

and the burst length and data rate are long-tailed (Pareto) variates. The parameters of the sampled distributions were fitted against recorded data of burst interarrival time (IAT), length, and size obtained from wideband CDMA networks.

5.1.2 | Mobility model

The mobility model takes 2 essential aspects of user mobility into account, namely, the distance between consecutive locations where the user resides for longer periods, and the fact that users—even though they occasionally make very long journeys—also tend to stay within a bounded area.[34,35]

The model used in the simulator is based on the observations made in González et al[29] that the radius of gyration, ie, the mean distance from a central point, and the locations a user visits over time can both be modelled by a truncated heavy-tailed distribution, and that the flight lengths displayed by a user are strongly correlated with its radius of gyration. On the basis of this observation, we sample heavy-tailed distances from a central position, which is randomly placed within the simulated area, but unique for each user, and an angle sampled from a uniform distribution to obtain goal positions for the next resting point. This results in heavy-tailed flight distances without excessively dispersing the users.

5.1.3 | Radio model

The radio model is based on 2 key components: a stochastic path loss calculation based on the model in Erceg et al[36] and a radio interference model developed by the authors. The path loss model is empirically based and emulates stochastic path loss per cell and UE location over a given distance, cell power, and antenna placement in one of 3 geographical types.

To calculate the radio resource demand of a user with a given bit rate demand and path loss, we estimate the signal to interference and noise ratio at the user's location. We assume the main components of this ratio to be the received signal power (RSP) of all (downlink) transmissions and calculate the noise and interference at a location as the sum of a fixed noise floor and the current RSP of all cells at the location. The output of the interfering cells is assumed to be cell power times the relative resource block utilisation averaged over a fixed time frame.

Handover decision is made based on smoothed downlink RSP measurements, offset by the current value of the CRE parameter, and updated once per second. The contribution of each connected UE to the load of its connected cell is a function of the bandwidth demand prescribed by the traffic model and the path loss at the UE's position.

5.2 | Simulation scenarios

For each scenario, the placement and effective range of a given number and type of cells are fixed. The movement and traffic events for each UE are reproduced over each run of the scenario by using a fixed seed for the random generator used to sample inter-event times and magnitudes.

Most UEs remain idle at any given time but maintain their mobility and traffic demand simulations, to generate connection and handover events at realistic rates.

Within each scenario, the effective range of each cell and path loss for each 5×5-m area is determined by the samples of 3 random variables in the path loss model. Two samples are chosen for each scenario and cell in addition to its placement and power, and one more is assigned to each area within a scenario.

5.3 | Collected statistics

We consider the following set of statistics within each of 4 classes of scenarios with fixed overall area size, number of cells, and UEs. Since placement and coverage of the cells vary between scenarios, we see a range of load levels and coverage between scenarios within a class. Each scenario is run for 30 simulated minutes, after a 3-minute run-in period. For each scenario size class, we generate between 50 and 100 scenarios and record for each time frame and scenario the following statistics:

1. The mean load over the cells
2. The max load over the cells
3. The mean of the 95th load percentiles for each cell, measured as a running average
4. The corresponding max over the 95th load percentiles
5. The Jain fairness index of over the cells for intracell running load mean and 95th percentiles
6. The proportion of requested bits served

7. The number of hand-ins per connected user over all cells

where the Jain fairness index,

$$J(x_1, \ldots, x_n) = \frac{\left(\sum_{i=1}^{n} x_i\right)^2}{n \sum_{i=1}^{n} x_i^2}, \tag{12}$$

measures the similarly of each metric (running load mean and 95th percentile), over the nodes i.

5.4 | Scenario classes

We summarise these statistics with quantiles and means for each metric over all time frames and scenarios within each class and vary the fixed area size, number of cells, and user as follows:

1. One macro and one smaller cell over an area of 2.25 km^2 and 7000 UEs
2. One macro and 2 smaller cells over an area of 4.0 km^2 and 8750 UEs
3. One macro and 3 smaller cells over an area of 6.25 km^2 and 10 500 UEs
4. Two macro cells and 4 smaller cells over an area of 9.0 km^2 and 17 500 UEs
5. Four macro cells and 11 smaller cells over an area of 27.0 km^2 and 34 900 UEs

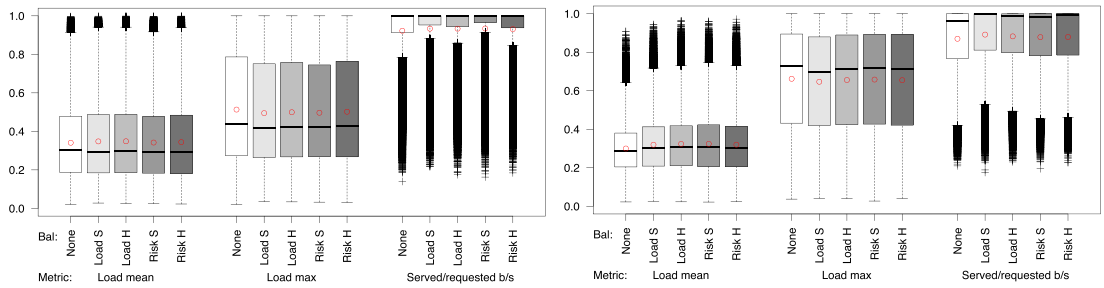
5.5 | Experimental setup

Experiments are performed with several variants of the load balancing method engaged and compared to a base case with CRE set to 0 for all cells. The first of these is the load balancing method described in Kreuger et al[1] (ie, with load mean metric and maximal scaling), and the second by mapping the distance between the actual and target loads to a given CRE range using the sigmoid mapping as described in Section 4.1.2. The third and fourth replaces the load metric of the 2 previous methods with risk of exceeding 95% of the cell capacity as described in Section 4.1.2.

The LTE standard allows CRE values in the range ±10 dB, but in practice, the impact on the signal quality of individual UEs attached to a cell with less than about 6 dB below the strongest one can be significant and should be avoided. This can be done in several ways, the simplest being a hard limit on the range of valid CRE values, but in principle, we can also use methods that ensure that such situations remain rare. We will report results for a CRE range of ±3 dB that guarantees that this will never happen, but also give examples for a CRE range of ±6 dB where of impact of and differences between the methods are more clearly seen. For the sigmoid mechanism, we will also report result on a ±10-dB CRE range with CRE and RSP difference statistics.

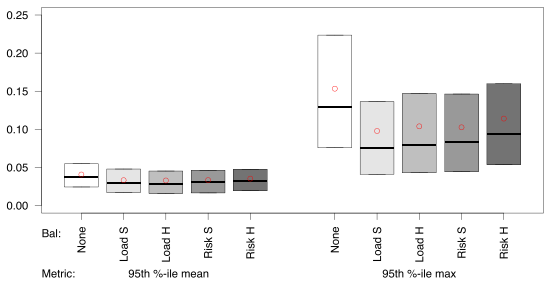
5.6 | Results

Let us have a look, first, at some general properties of the different classes of scenarios. Figure 4 shows box plots over the quantiles and mean of 3 metrics: (1) Mean and max momentary loads over the involved nodes and (2) the proportion of the requested bits transferred. In this case, load means remain approximately constant over differently sized scenarios, while load max increases significantly and system efficiency goes down somewhat as the number of cells (and users) increases. In all cases, we see small but clear improvements for all the balancing methods with respect to the base line.

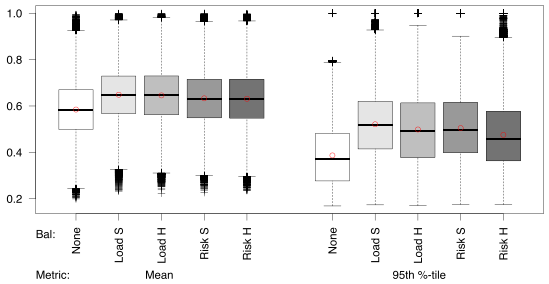


(A) Two-cell scenarios. (B) Four-cell scenarios.

FIGURE 4 General trends. Means over 100 scenarios, each lasting 30 minutes, ±3-dB CRE range. CRE, cell range expansion

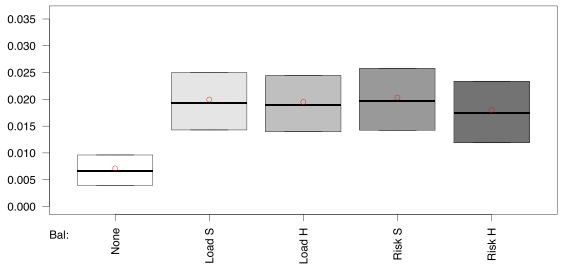


(A) Mean and max of the 95th %-ile over the nodes.

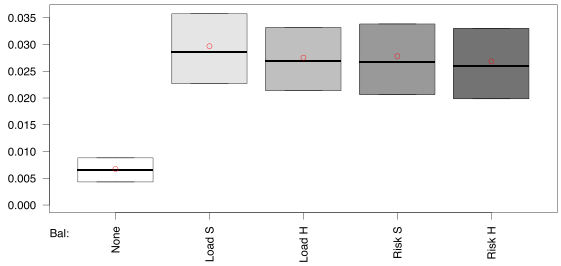


(B) Jain fairness index of mean and 95th %-ile over the nodes.

FIGURE 5 95th percentile metrics. Means over 100 six-cell scenarios, 30-minute runs, ±3-dB CRE range. CRE, cell range expansion

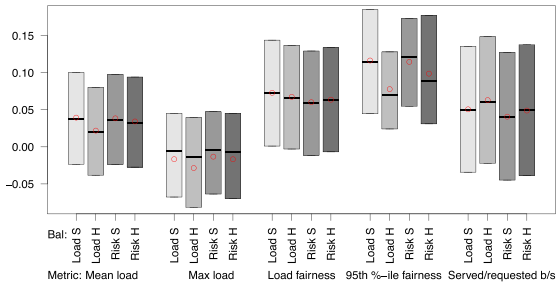


(A) 6-cell scenarios,±3 dB CRE range.

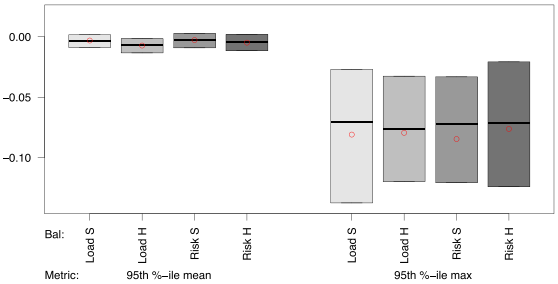


(B) 15-cell scenarios, ±6 dB CRE range. High sigmoid scale.

FIGURE 6 Handover frequency in $^{\text{handovers}}/_{s}$ per connected user. Means over 100 and 50 scenarios, 30-minute runs



(A) Mean, max and fairness over nodes & system throughput.



(B) Mean and max 95th %-iles over nodes.

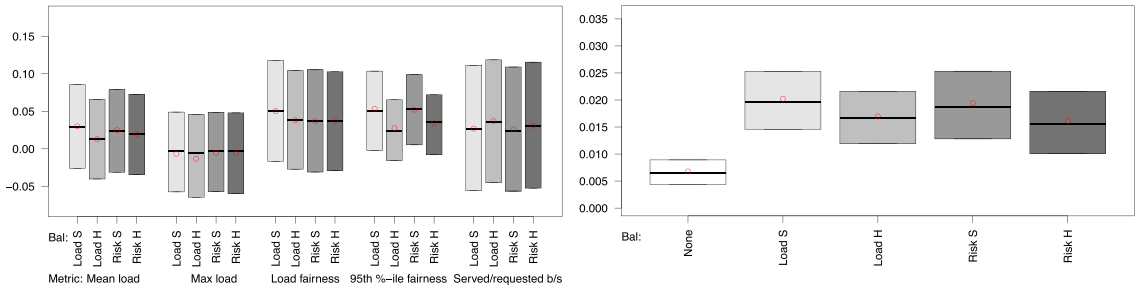
FIGURE 7 Relative improvement vs baseline. Means over 50 15-node, 30-minute scenarios, ±6-dB CRE range. CRE, cell range expansion

The next set of graphs (Figure 5) shows more clearly the impact of the various methods. Figure 5A shows the mean and max over the nodes of the 95th load percentile, while Figure 5B shows the corresponding Jain fairness index of the load means and 95th load percentiles, respectively. Note the strong reduction of the 95th percentile maximum, ie, that of the highest loaded node, and the corresponding improvement of the Jain index, especially for the 95th percentile.

This improvement comes at the cost of an increased number of handovers. Figure 6 shows how the handover frequency more than doubles for the maximal scaling mechanism, while the cost for the risk balancing, in particular the one using the sigmoid mechanism, is more moderate.

5.6.1 | Method comparisons

Next, we will see how the methods compare. Figure 7 shows the effect of each method relative to the base line for all metrics discussed above for the 15-cell, 34 900 UE case and with a ±6-dB CRE range. In Figure 7A, we see that the mean load goes up by between 2% and 4%, but that the max, ie, the load of the highest loaded node, simultaneously goes down by between 1.5% and 3%. The increase of the mean load is matched by a corresponding increase of served bits by between 4.1% and 6.5%. Mean load fairness is improved by around .07 while the fairness of the really high loads (95th load percentile) is increased by between .08 and .12. In Figure 7B, we see a slight decrease in mean of the 95th load percentile, but a decrease of around 8% in the 95th percentile of the highest loaded node over all times and runs.



(A) Mean, max and fairness over nodes & system throughput. (B) Handover freq. in $\mathrm{handovers}/s$ per connected user.

FIGURE 8 Relative metrics vs base line. Means over 50 15-node, 30-minute scenarios, ±3-dB CRE range. CRE, cell range expansion

These numbers are for a CRE range of ±6 dB and the sigmoid parameters set to more or less match the performance of the maximal scaling mechanism. Looking at Figure 6B, the cost in terms of handovers shows an increase from $\frac{.4}{min.}$ for the base line to $>\frac{1.7}{min.}$ for all balancing methods. Such numbers are probably not acceptable in practice, so we will next examine a more realistic case. The ones presented so far do show that (1) the level of high load reduction achievable by this type of method and (2) the sigmoid mechanism can be parametrised to reproduce the performance of the maximal scaling mechanism, as long as we disregard the cost.

In Figure 8, we see the corresponding results for a CRE range of ±3 dB and sigmoid parameters set to achieve a more reasonable balance between balancing performance and handover cost. In Figure 8A, we see a modest reduction of around 1% in the occurrence of high loads. The sigmoid scaling of the mean load (S) is most efficient at 1.3%, but even the sigmoid scaling of the high load risk comes in at 0.5%. The load fairness gives a detailed view, with improvements in Jain index values between .04 and .05. The difference between the methods is most visible in the case of the Jain index over the running 95th load percentile, which are improved by 0.27 to 0.53. We can also see that the sigmoid scaling appears to improve system throughput more than the maximal scaling. In Figure 8B, we see that by adjusting the sigmoid parameters, we can achieve comparable results to the maximal scaling but at a reduced cost in terms of handovers. Here, the cost in handover frequency for the risk balancing with sigmoid scaling is only increased to $\frac{1.20}{min.}$ from $\frac{.41}{min.}$ for the baseline. Both sigmoid variants stay below $\frac{1.0}{min.}$

5.6.2 | Sigmoid scaling with wide CRE range

We will examine one more example where we reduce the sigmoid scale further, but at the same time increase the CRE range to that allowed by the LTE standard, ie, ±10 dB. This means that there is a non-zero risk of an individual UE connecting to a node with a RSP up to 20 dB lower than the best one at a given location. In Figure 9A-D, we plot the distribution of differences between that of the node the UE is connected to and that of the one with the strongest signal at the location of the UE for 50 scenarios of 3 nodes with low scale parameters and ±10-dB limit, the sigmoids.

Since the individual handover decisions are actually based on a running estimate of the measured RSP value modulo the CRE offset values, the CRE difference is not directly translatable to RSP difference. We therefore also measured the resulting difference between the momentary RSP of the connected node and that of the one with the strongest signal, with the result shown in Figure 9E-H. It turns out that the probability of connecting to a node with a RSP lower than the one with the strongest signal is actually lower using the sigmoid over the full ±10-dB range than using the maximal scaling mechanisms with an ±3-dB range. The reason for this is that the maximal scaling forces UEs to connect to a nonoptimal node even if the metric difference relatively is small, while the sigmoid uses large CRE values only when then metric difference is also large. Note that the bin containing the zero difference is truncated in the plots and the absolute density values are very low. For the vast majority of UE time frames, the UE connects to the strongest signal and reports a zero value. Note also that both risk balancing methods reduce the impact compared to the mean load balancing ones.

6 | DISCUSSION

6.1 | Experimental highlights

It is clear from the results of Section 5.6 that all versions of the proposed balancing method have significant and positive effect on the metrics examined, but there are also significant differences between the methods. Balancing the risk metric,

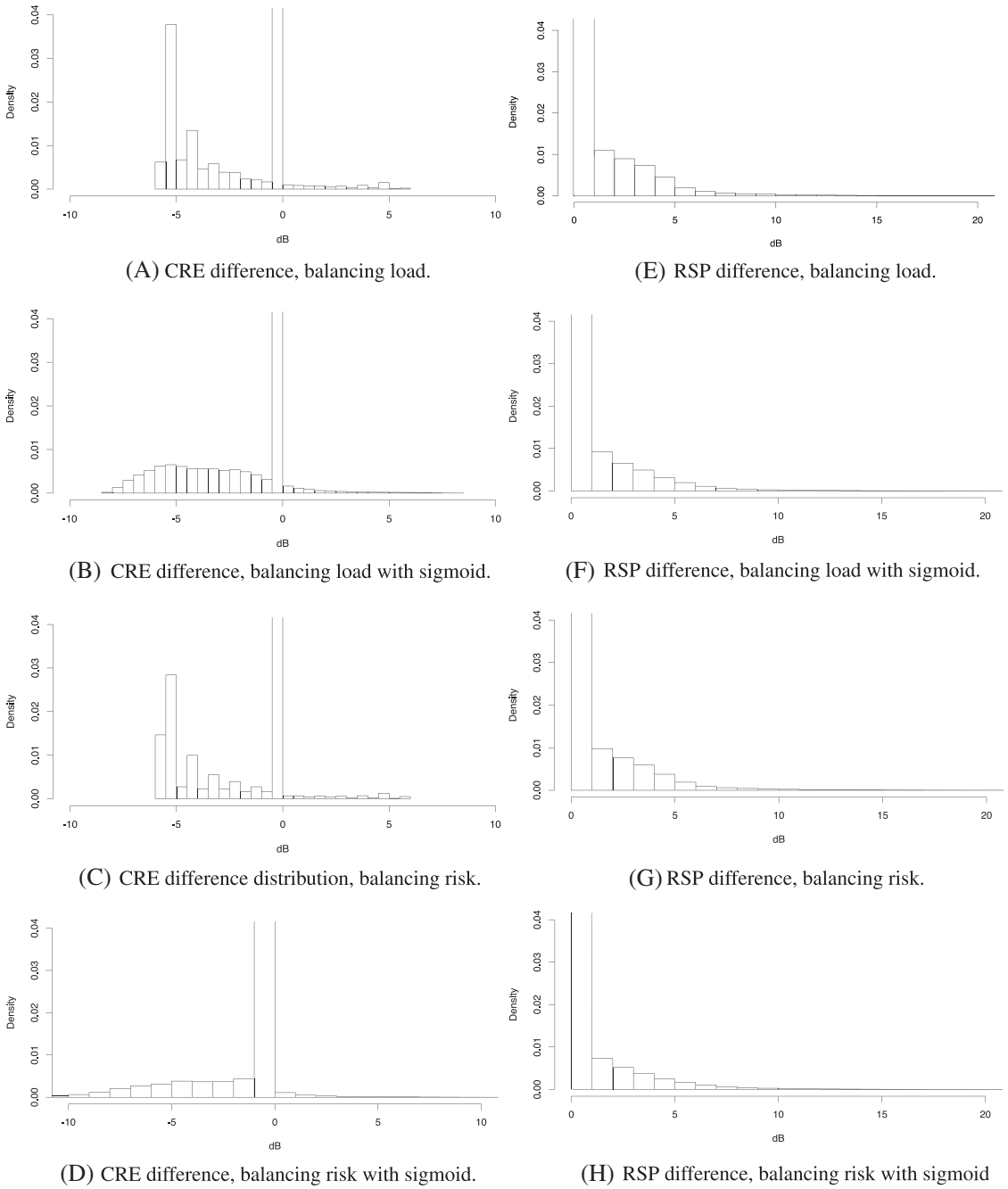


FIGURE 9 Cell range expansion (CRE) and received signal power (RSP) difference distributions for each method. Means over 50 3-node 30-minute scenarios of ±10-dB CRE range for the sigmoids and ±3 dB for the others

as expected, achieves more balance between the really high loads, as seen in the 95th percentiles, and a bit less balance overall, but this is also reflected in lower cost in terms of handovers.

Highly significant reductions in handover cost can be achieved by using the sigmoid mapping, especially if we use low scaling factors and allow rare cases of extreme imbalance to result in occasional CRE offsets larger than 6 dB. As indicated by Figure 9, the impact in terms of channel quality may not be so significant in practice. Finding practical sigmoid scaling factors and CRE ranges in a deployed RAN will require further experimentation, of course.

6.2 | Modelling considerations

There is a risk that the load distributions will not turn out to be log-normal for every type of resource and RAN. In such cases, the distribution may be modelled by other parametric distributions or estimated empirically using quantiles. To the

extent that the underlying demand distribution indeed *is* log-normal, a more thorough comparison between the modelled tail mass and measurements of the secondary mode (see Section 3) at high loads is also needed. However, the simplicity of the log-normal model, and the straightforward extraction of modelled quantiles proposed in Section 4.1.1, makes it attractive for many cases where the underlying process is even approximately log-normal.

6.3 | Implementation aspects across different RATs

In Section 4.2, we outlined a version of the method for WiFi that is very similar to the one described for LTE in the context of intra-RAT load balancing. In practice, the implementation of DTC mechanisms and mobility management functions for both intra-RAT and inter-RAT operations is specific to the underlying technology and equipment. For intra-RAT operations, the implementation of the actuation mechanisms may therefore look very different depending on the RAT at hand. In cases of inter-RAT operations across multiple heterogeneous RATs, the key issue is to ensure that computed targets and biases are comparable, which may require additional processing.

Even though we do not present any results for inter-RAT load balancing, we maintain that the algorithm *is* RAT-agnostic, due to the generality of the method and the use of probabilistic risk estimates favouring control in terms of high-level abstractions. In cases involving multiple and heterogeneous RATs, the level of implementation efforts varies with the diversity of the underlying technologies. In systems with similar RATs operating in line with more or less common standards and functionality, RAT-specific implementations should be relatively easy to adapt and modify with little or moderate efforts. Load balancing within a cellular system involving different generations of the same type of RAT is a typical example.

The expected densification of radio nodes needed for maintaining service availability in the future requires that several technical challenges are solved for bridging the gap between heterogeneous infrastructures to achieve effective mobility operations. For example, the case of handoffs between cellular and WiFi comes with its own set of open problems, even though standards in this domain appear to be converging.[37-39] Once mechanisms for smooth handoffs between WiFi and cellular (and other RATs) are in place, the method proposed in this paper can be widely applied and extended for balancing one or multiple metrics within and across various RATs.

6.4 | Combining DTC and centralised control

The formulation of the mechanism as a distributed algorithm enables a range of implementation possibilities from completely distributed, with only sparse monitoring information and control parameter adjustments passed between controllers and nodes, to a physically centralised solution where high-resolution measurement is regularly collected and centrally managed. Regardless of the deployment strategy, monitoring data can be aggregated to a centralised monitoring function, which is a highly useful feature inherent to the design of the method, offering great implementation and operational flexibility. Furthermore, the self-adjusting balancing method offers the right type of high-level policy parametrisation to fit into a centralised management scheme through the use of probabilistic metrics and thresholds. More specifically, the sigmoid scaling and range parameters, as well as the chosen load level for the risk estimates, are suitable candidates for a controller API, enabling policy adjustments to shift balancing performance against handover and channel quality costs.

The overall idea of a logically centralised RAN control is effective coordination of radio resources. We believe that effective (ie, timely and scalable) coordination of network resources can be realised in a distributed system that implements both logically centralised control functions while operating in a self-organising manner, where applicable. As an example of the combination of DTC and centralised management, we envision the proposed metrics and load balancing approach to be used in a programmable RAN setting, involving a programmable infrastructure controlled by a logically centralised controller and a high-level framework for managing services and network operations (Figure 10). In this setting, estimated overload risks are communicated to the controller and management levels for sophisticated coordination and control of heterogeneous radio resources and parameters in relation to network operation and service requirements. For load balancing, actuator thresholds and scaling and range parameters are here communicated and set locally in the involved nodes and used for triggering RAT-specific and self-organised load balancing. In a logically centralised controller setting, the probabilistic metric abstractions are necessary components for providing a unified network view while facilitating infrastructure configurations as well as communication between controllers for eventually self-organised inter-RAT operations.

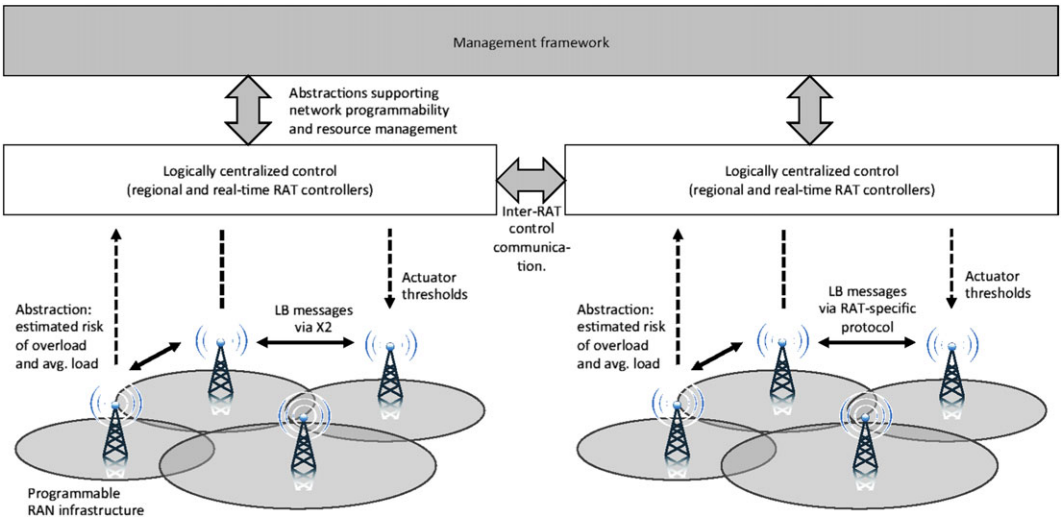


FIGURE 10 Conceptual overview of load balancing and corresponding abstractions in a programmable RAN

7 | CONCLUSIONS

We have described a generic method for dynamically balancing a given resource metric in distributed systems and exemplified its applicability in RANs with focus on load balancing in LTE and WiFi. The main features of the method include simple deployment by the use of already existing mechanisms available in radio access technologies and standards; local and low complexity modelling; flexible implementation for both distributed and centralised management; and probabilistic metrics and targets supporting RAT-agnostic operation and control.

The method has successfully been evaluated in a compound simulation environment based on a cellular RAN with path loss, traffic, and mobility models. The results indicate that load balancing based on probabilistic overload risk metrics, compared to load balancing based on means, offers more robust solutions with the benefit of, eg, fewer handovers. The use of sigmoids for mapping target metrics to CRE bias values also provides a simple and convenient way of controlling the trade-off between balancing performance and cost.

The method is currently limited to a single metric, and a future and useful extension would be the capability of balancing multiple metrics or composite targets. The applications enabled by such an extension would in a RAN setting entail balancing of, eg, backhaul and node compute capacity in addition to radio resources (eg, for RAN slicing), as well as flexible coverage patterns for radio access nodes. Finally, the general properties of the DTC approach open up for solving resource management problems also in wired networks, such as core networks and cloud systems, with applications ranging from routing, resource allocation, and load balancing.

ACKNOWLEDGMENTS

This work was funded in part by the Swedish Foundation for Strategic Research (reference number RIT15-0075) and by the Commission of the European Union in terms of the 5G-PPP COHERENT project (grant agreement no. 671639) within the H2020 LEIT Information and Communication Technologies program.

ORCID

Per Kreuger http://orcid.org/0000-0002-9331-0352

REFERENCES

1. Kreuger P, Görnerup O, Gillblad D, Lundborg T, Corcoran D, Ermedahl A. Autonomous load balancing of heterogeneous networks. In: 2015 IEEE 81st Vehicular Technology Conference (VTC Spring); 2015; Glasgow. 1-5.
2. Kreuger P, Gillblad D, Görnerup O, Lundborg T, Corcoran D, Ermedahl A. Methods, nodes and system for enabling redistribution of cell load. *Patent PCT/SE2014/050790*; 2016. Ericsson AB.
3. Gil J. Renaming and dispersing: techniques for fast load balancing. *J Parallel Distrib Comput*. 1994;23(2):149-158.

4. Bonomi F, Kumar A. Adaptive optimal load balancing in a nonhomogeneous multiserver system with a central job scheduler. *IEEE Trans Comput*. 1990;39(10):1232-1250.
5. Kameda H, Li J, Kim C, Zhang Y. *Optimal Load Balancing in Distributed Computer Systems*. London: Springer Verlag; 1997.
6. Kim C, Kameda H. An algorithm for optimal static load balancing in distributed computer systems. *IEEE Trans Comput*. 1992;41(3):381-384.
7. Hui CC, Chanson ST. Improved strategies for dynamic load balancing. *IEEE Concurrency*. 1999;7(3):58-67.
8. Campos LM, Scherson I. Rate of change load balancing in distributed and parallel systems. *Parallel Comput*. 2000;26(9):1213-1230.
9. Corradi A, Leonardi L, Zambonelli F. Diffusive load-balancing policies for dynamic applications. *IEEE Concurrency*. 1999;7(1):22-31.
10. Chen XH. Adaptive traffic-load shedding and its capacity gain in CDMA cellular systems. *IEEE Proceedings - Commun*. 1995;142(3):186-192.
11. Bahl P, Hajiaghayi MT, Jain K, Mirrokni SV, Qiu L, Saberi A. Cell breathing in wireless lans: algorithms and evaluation. *IEEE Trans Mob Comput*. 2007;6(2):164-178.
12. Du L, Bigham J, Cuthbert LG, Nahi P, Parini C. Intelligent cellular network load balancing using a cooperative negotiation approach. In: WCNC; 2003; Miami. 348-353.
13. Lobinger A, Stefanski S, Jansen T, Balan I. Load balancing in downlink LTE self-optimizing networks. In: Vehicular Technology Conference (VTC 2010-Spring), 2010 IEEE 71st; 2010; Taipei. 1-5.
14. Siomina I, Yuan D. Load balancing in heterogeneous LTE: range optimization via cell offset and load-coupling characterization. In: 2012 IEEE International Conference on Communications (ICC); 2012; Kansas City. 1357-1361.
15. Wang H, Ding L, Wu P, Pan Z, Liu N, You X. Dynamic load balancing and throughput optimization in 3GPP LTE networks. In: Proceedings of the 6th International Wireless Communications and Mobile Computing Conference, IWCMC '10. ACM; 2010; New York, NY, USA:939-943.
16. Hao W, Nan L, Zhihang L, Ping W, Zhiwen P, Xiaohu Y. A unified algorithm for mobility load balancing in 3GPP LTE multi-cell networks. *Science China*. 2012;1(12).
17. Foukas X, Nikaein N, Kassem MM, Marina MK, Kontovasilis K. FlexRAN: a flexible and programmable platform for software-defined radio access networks. In: Proceedings of the 12th International on Conference on Emerging Networking Experiments and Technologies, CoNEXT '16. ACM; 2016; New York, NY, USA:427-441.
18. Gudipati A, Perry D, Li LE, Katti S. SoftRAN: software defined radio access network. In: Proceedings of the Second ACM Sigcomm Workshop on Hot Topics in Software Defined Networking. Hong Kong: ACM; 2013:25-30.
19. Kostopoulos A, Agapiou G, Kuo FC, et al. Scenarios for 5G networks: the COHERENT approach. In: 2016 23rd International Conference on Telecommunications (ICT). Thessaloniki: IEEE; 2016:1-6.
20. Nunes BAA, Mendonca M, Nguyen XN, Obraczka K, Turletti T. A survey of software-defined networking: past, present, and future of programmable networks. *IEEE Commun Surveys Tutor*. 2014;16(3):1617-1634.
21. Kreuger P, Gillblad D, Arvidsson Å. Zero configuration adaptive paging (zCap). In: IEEE 76th Veh. Technol. Conf. IEEE; 2012; Québec. http://www.ieeevtc.org/vtc2012fall/, 978-1-4673-1881-5.
22. Kreuger P, Gillblad D, Arvidsson Å. zCap: a zero configuration adaptive paging and mobility management mechanism. *J Network Manage*. 2013;23:235-258.
23. 3GPP Technical Specification Group: Radio Access Network. X2 protocol. Technical Report TS36.423 (Release 12), 3GPP; 2013.
24. Bishop CM. Pattern recognition. *Machine Learning*. 2006;128:1-58.
25. Fukuda K. Towards modeling of traffic demand of nodes in large scale network. In: ICC'08. IEEE International Conference on Communications, 2008. Bejing: IEEE; 2008:214-218.
26. Downey AB. Lognormal and pareto distributions in the internet. *Comput Commun*. 2005;28(7):790-801.
27. Kreuger P, Steinert R. Scalable in-network rate monitoring. In: IFIP/IEEE Integrated Network Management — IM'15. IFIP/IEEE; 2015; Ottawa, Canada. 866-869.
28. Naboulsi D, Fiore M, Ribot S, Stanica R. Large-scale mobile traffic analysis: a survey. *IEEE Commun Surveys Tutor*. 2016;18:124-161.
29. González MC, Barabási AL, Hidalgo CA. Understanding individual human mobility patterns. *Nature*. 2008;453:779-782.
30. IEEE. 802.11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. IEEE Standard 802.11, IEEE-SA 5 April 2012 revision, https://doi.org/10.1109/IEEESTD.2012.6178212
31. Suresh L, Schulz-Zander J, Merz R, Feldmann A, Vazao T. Towards programmable enterprise WLANS with Odin. In: Proceedings of the First Workshop on Hot Topics in Software Defined Networks. Helsinki: ACM; 2012:115-120.
32. Schulz-Zander J, Mayer C, Ciobotaru B, Schmid S, Feldmann A. OpenSDWN: Programmatic control over home and enterprise WiFi. In: Proceedings of the 1st ACM SIGCOMM Symposium on Software Defined Networking Research. Santa Clara, CA, USA: ACM; 2015:16.
33. Riggio R, Rasheed T, Granelli F. Empower: a testbed for network function virtualization research and experimentation. In: IEEE Software Defined Networks for Future Networks and Services 2013 - SDN4FNS. Trento, Italy: IEEE; 2013:1-5.
34. Song C, Koren T, Wang P, Barabási AL. Modelling the scaling properties of human mobility. *Nat Phys*. 2010;6:818-823.
35. Song C, Qu Z, Blumm N, Barabási AL. Limits of predictability in human mobility. *Science*. 2010;327(5968):1018-1021.
36. Erceg V, Greenstein LJ, Tjandra SY, et al. An empirically based path loss model for wireless channels in suburban environments. *IEEE J Sel Areas Commun*. 1999 July;17(7):1205-1211.

37. IEEE. 802.11u: Amendment 9: Interworking with External Networks. IEEE Standard 802.11, IEEE-SA 5 April 2011. http://standards.ieee.org/about/get/802/802.11.html.
38. IEEE. 802.21: Media Independent Handover. IEEE Standard 802.21, IEEE-SA 2015. http://standards.ieee.org/about/get/802/802.21.html.
39. ETSI. Universal mobile telecommunications system (UMTS); LTE; IP flow mobility and seamless wireless local area network (WLAN) offload. TS 123 261, ETSI 2012.

Per Kreuger is a senior researcher employed at the Swedish Institute of Computer Science (Rise SICS) since 1986 and has been active throughout his career as both project and group leader, but more recently mostly as technical specialist. He earned his PhD in Computer Science at the University Gothenburg in 1995, with early research focus within computational logics and optimisation, but has more recently focused on statistical modelling and distributed algorithms, with applications in network management and traffic analysis.

Rebecca Steinert is a senior research scientist and driver of ICT research at Rise SICS AB (employed since 2006). Her research group offers expertise in system-oriented solutions based on applied machine learning and data analytics for management of software-defined and virtualised networking infrastructures. From KTH Stockholm, she has a BSc in real-time systems (2002), an MSc in autonomous systems and machine learning (2008), and a PhD (2014) in distributed and probabilistic network management.

Olof Görnerup is currently a senior researcher at the Swedish Institute of Computer Science (SICS) in Stockholm. He received a MSc, a PhLic, and a PhD in complex systems from Chalmers University of Technology in Sweden in 2003, 2007, and 2008, respectively. During 2002 to 2004, he was a Graduate fellow at the Santa Fe Institute in New Mexico, USA. His current research interests revolve around both fundamental machine learning, data mining and data analytics, and their applications.

Daniel Gillblad holds a MSc in Electrical Engineering and a PhD in Computer Science, both from the Royal Institute of Technology (KTH) in Sweden. He is currently the director of the Decisions, Networks and Analytics (DNA) laboratory at the Swedish Institute of Computer Science (SICS). The laboratory performs research within machine learning, data analytics, networked intelligence, scheduling and optimisation, and their applications. His research interests are currently focused around graph- and probabilistic methods for large-scale data analytics and machine learning, network management, diagnostics, and mobility modelling.

How to cite this article: Kreuger P, Steinert R, Görnerup O, Gillblad D. Distributed dynamic load balancing with applications in radio access networks. *Int J Network Mgmt*. 2018;28:e2014. https://doi.org/10.1002/nem.2014