Autonomous load balancing of heterogeneous networks

Per Kreuger, Olof Görnerup and Daniel Gillblad

Decisions, Networks and Analytics (DNA) Laboratory

Swedish Institute of Computer Science (SICS Swedish ICT), SE-164 29 Kista, Sweden

Email: {piak, olofg, dgi}@sics.se

Tomas Lundborg, Diarmuid Corcoran and Andreas Ermedahl

Development Unit Radio, Systems and Technology

Ericsson AB, SE-164 80 Kista, Sweden

Email: {tomas.lundborg, diarmuid.corcoran, andreas.irmedahl}@ericsson.com

Abstract—This paper presents a method for load balancing heterogeneous networks by dynamically assigning values to the LTE cell range expansion (CRE) parameter. The method records hand-over events online and adapts flexibly to changes in terminal traffic and mobility by maintaining statistical estimators that are used to support autonomous assignment decisions. The proposed approach has low overhead and is highly scalable due to a modularised and completely distributed design that exploits self-organisation based on local inter-cell interactions. An advanced simulator that incorporates terminal traffic patterns and mobility models with a radio access network simulator has been developed to validate and evaluate the method.

Index Terms—autonomous network management; self-organising heterogeneous networks; distributed algorithms; statistical modelling

I. INTRODUCTION

Heterogeneous networks (HetNets) refer to radio access networks (RANs) where several types of network nodes – differing with respect to transmission power, radio bandwidth, backhaul capacity, placement etc. – interact to provide network access and communication services to a set of terminals referred to as user equipments (UEs). Managing HetNets is highly challenging due to their complex and dynamic topology involving both macro- and low-power cells, where the latter may be added and removed as required by local conditions such as temporary hotspots and flash crowds. This makes manual management very cumbersome and inefficient at best. In particular, the traditional mechanism used to allocate UEs to nodes, which is based on the relative *signal to noise ratio* (SINR) for candidate nodes at the UE's location, is no longer sufficient [1]. The smaller nodes are expected to off-load the larger ones for nearby and relatively stationary UEs with high bandwidth demand, but their transmission power is generally not sufficient to dominate (in terms of relative SINR) over the signals of adjacent larger nodes (i.e. with higher transmit power). There are mechanisms in modern standards to cope with this problem, but the dynamicity of the problem makes manual management of the parameters controlling these mechanisms unwieldy.

To enable scalable, robust and autonomous management of HetNets, we argue that we instead need to rely on automated,

distributed, adaptive and data-driven mechanisms based on self-organisation. In this paper we will present a distributed on-line mechanism to autonomously assign target load values for each involved node based on the load situation in its environment, and use these and estimates of actual loads as input to a mechanism to redistribute and balance load within the network. Once target loads have been determined, several ways to use this information to achieve improved load balance can be envisioned, but we will describe one particular approach based on determining suitable values for a bias parameter used in the calculation of relative SINR values reported by the UEs at handovers. In LTE, such a mechanism is provided in the form of a *cell range expansion* (CRE) parameter, which is associated with each node (or pair of nodes), and intended to bias the SINR calculation performed by the UEs towards smaller, and/or less loaded nodes.

The prevalent solution for this problem in currently deployed systems is to manually configure the CRE parameter based on expected network load in a given area. This may be feasible for situations where load, node placement and interference are fairly static, but in many future scenarios, this will *not* be the case. For example, small nodes may be added without prior planning or direct network operator control over exact placement, and where UE traffic demand and mobility may vary widely on both short (seconds and minutes) and long (days) time scales. Manually configuring these parameters of networks in such scenarios may not be a viable alternative.

A. Related work

Load balancing for cellular networks has been well studied and the general idea to use measurements of the current load distribution in the network to do so is certainly not new. For example, [2] introduces a method based on integer programming to assign CRE values to each node, given load levels of the entire network. Since this method is centralised it 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, which only then can be redistributed to the nodes of the network. It is unclear how the authors intend to handle the delays and

scalability issues implied by such a mechanism. Similar issues arise in an approach described in [3] which is also centralised, but uses enforced handovers rather than manipulation of the CRE parameter. In contrast, our proposal is distributed and localised in the sense that the decision of which CRE value to use is set by each node, albeit after exchanges with other nodes in its immediate proximity.

The proposal described in [4] uses an estimate of the remaining available capacity of each node to assign CRE values for pairs of nodes based on interactions between eNodeBs on the X2 interface [5], specifically the *TNL Load Indicator* and the *Composite Available Capacity* (CAC) messages. The load indicator is very coarse (2 bits) and used only to determine which nodes should participate in the balancing negotiations, while locally determined CAC values are calculated using a *fixed* target load value for each node. Pairwise CRE values are then computed by scaling CAC ratios with operator specific parameters. Using fixed target load implies an imperfect adaptation to variations in load distributions and a separate heuristic is employed to determine when, and for which nodes, the proposed mechanism should be triggered.

Lobinger et al. [6] report work wherein load balancing is performed by considering individual UE “satisfaction” based on both SINR and node radio load. They propose a mechanism based on selecting individual UE candidates and target cells for off-loading overloaded cells, evaluating the effect on total “satisfaction”, before committing to each handover, and adjusting CRE values to ensure that the UE stays put in the new cell. They verify their results using sophisticated RAN and mobility simulations, but the scenarios they consider are based on uniformly placed macro-nodes, and although their proposed mechanism is advertised as autonomous, it is not clear how well it scales, considering its relative complexity.

In our method, the target loads are dynamically determined using a distributed algorithm that (potentially) involves all nodes. This approach is less dependent on ad hoc, and possibly arbitrary, parameters, and should improve both scalability and performance under varying load conditions. We note also that the simulations reported in [4] does not model UE mobility and use only constant UE traffic demands, while our method has been evaluated in simulations, which include both realistic traffic variations and a range of UE mobility patterns.

B. Method

We will present our approach for autonomous RAN load balancing using one particular way of manipulating the CRE parameter as it appears in LTE. We will start by focusing on the calculation of *target loads* which we consider the core result of this work. On the most basic level, the idea is to set the *target load value* of each node to *the average of its current load estimate and the targets load values in its environment*. How to select which nodes to explicitly include in this computation, and how to weight the influence of each of the nodes in a given environment is however non-trivial.

The solution we propose is exemplified in the RAN load balancing application, where we create and dynamically up-

date a list representing a neighbourhood of adjacent nodes to and from which any given node has hand-overs. Each node will query the nodes in its neighbourhood of their *target* loads, and set its own target load to the average of those in its neighbourhood, and its own current load. Since the target values of adjacent nodes will in general be mutually dependent, this computation may be iterated until a local equilibrium is reached. Here we propose an approach which converges under stable conditions and adapts quickly and robustly under realistic variations of load and UE mobility.

One particularly attractive property of this method is that, as long as a majority of the nodes computes their target loads in the same way, each node will in effect be implicitly influenced by the entire network, but with the influence of any other node scaled by a factor depending on its distance.

1) *Local neighbourhoods*: Each node maintains a list, which we refer to as its *local neighbourhood*, of nearby nodes with which it has overlapping coverage. This can be dynamically updated by estimating the probability of a handover to and from each other node and by including the most likely nodes until a threshold for the probability mass represented by the chosen nodes is reached. The handover probability estimator can e.g. be based on a discrete Bayesian estimation scheme, very similar to the one used in [7], [8], but here based on handover events.

2) *Target load updates*: This update procedure is the core of the method. What is described here is the complete procedure to update a single node, but the claimed properties of the mechanism assumes that the procedure applied in every node.

Whenever a node i detects a sufficiently large change in its current load estimate l_i , it executes the following procedure:

- 1) Retrieve, for each node j in the neighbourhood n_i of i , its current *target load* t_j .
- 2) Set the *target load* t_i of i , to the mean of i 's current load estimate l_i and the target loads in n_i :

$$t_i = \frac{1}{1 + |n_i|} (l_i + \sum_{j \in n_i} t_j). \quad (1)$$

- 3) Since all the nodes in n_i calculate *their* target loads in the same way, possibly using t_i as input, nodes in n_i are requested to recalculate their target load values using *their* current local neighbourhoods. Once this calculation is complete, node i recalculates *its* target load t_i , using the updated target loads of the nodes in its neighbourhood. This procedure is iterated until the difference between two successive calculations of t_i reaches a given threshold.

The procedure will terminate as long as the loads and neighbourhoods n_j of the nodes in the neighbourhood n_i of the node i remains approximately stable at the (short) time-scale at which the iterations are performed. If this is not the case, we can limit the number of iterations to a fixed upper bound.

a) *One step target load adjustment*: Steps 1-2 above is illustrated in more detail by the right flow chart of Figure 1. That is, upon receiving the message `adjustTarget` a node

Fig. 1. Flow charts for the updateTarget and adjustTarget actions.

in the state *wait*, performs one iteration of the target load update using an estimate (e.g. a running mean) of its current load (in terms of e.g. downlink radio bandwidth saturation), and the target loads of all its neighbours, stores its results and returns to the *wait* state.

b) Neighbourhood wide target load update: The complete target load update routine, including the iteration of step 3 above is illustrated in the left flow chart of Figure 1. That is, on receiving message *updateTarget* a node in the state *wait*, first performs a local adjustment by calling *adjustTarget* and then iterates over its neighbourhood, requesting each node to adjust *its* target load. Assuming that the local target load adjustment is atomic, the original node then again updates its own target load and if the new value differs sufficiently from the previous one, the procedure is repeated until a neighbourhood-wide equilibrium (or a maximum number of iterations) is reached.

Note that the *adjustTarget* and *getTarget* requests and their responses in the flow charts are messages that need to be passed between the nodes via a node to node interface when implemented in this completely distributed fashion.

3) Load balancing: One use of target loads calculated as above, is to assign a *CRE* value to each node with the goal to balance the load of the nodes of the system.

To redistribute the load of the network towards the load distribution represented by the calculated target loads, we can assign to each node a *CRE* value o_i in a suitable range (e.g. $[0, 6]$ dB) to i , which maximises its likelihood to achieve its target load t_i . We propose to do this by calculating, for each node i , the minimum $\hat{d}$ and maximum $\bar{d}$ target load to (actual) load differences $t_k - l_k$ for $k \in \{i\} \cup n_i$. This gives us a range of differences $[\hat{d}, \bar{d}]$ in the neighbourhood of i which we can use to scale the corresponding local difference $p_i - l_i$.

$$o_i = \frac{p_i - l_i}{\bar{d} - \hat{d}} 6dB \quad (2)$$

This has the advantage of using the entire range of *CRE*

Fig. 2. Flow chart for the updateOffset action (left) and the main entry point (right) for the local update.

values available locally, but tends to give large swings as the maximum difference in the neighbourhood approaches zero. In our current implementation we reduce this tendency by using a cutoff ϵ on $\hat{d} - \bar{d}$, beyond which we avoid adjusting the *CRE* value. The flow charts of Figure 2 shows this procedure and the main entry point to the local update in more detail. Although the relatively simple approach presented here appears to work well in our experiments, other mechanisms to manipulate the *CRE* values (or other mechanisms e.g. forced handovers) to reach the ideal represented by the target load are also possible.

II. MODEL SYSTEM

The proposed method has been evaluated in a simulator that uses the following model components:

A. Traffic model

The traffic model is based on a collection of random processes generating (downlink) traffic bursts. The simulator maintains one such process for each 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 data of burst inter-arrival time (IAT), length and size obtained from HSPA networks.

B. Mobility model

The mobility model takes two 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 [9]–[11].

The model is based on the observations made in [12], that the radius of gyration (RoG) – i.e. the mean distance from a central point and the locations a user visits over time – can be modelled by a truncated heavy-tailed distribution (THT), and that the flight lengths displayed by a user is strongly correlated with its RoG. Based on this observation, we sample

heavy-tailed distances from a central position, which is unique for each user, and use 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.

For the simulations and experiments reported here we have chosen user UE centre positions according to a normal distribution with a large variance, but confined to fall within a restricted area. The users themselves can move outside the area, but will tend to return to the general area of their centre point. This generates UE moves and rests which are similar to Lévy walks, but non-dispersing, even over long simulations.

C. Radio model

The radio model is based on two key components; a stochastic path loss calculation based on the result in [13], and a simple interference model developed by the authors.

The path loss model is empirically based and emulate stochastic loss over a given distance, cell power and antenna placement height in one of three geographical types. It is not ideally adopted for use in urban areas, but nevertheless approximates typical path loss and variance at given UE positions.

The interference model calculates the noise at a given location as the sum of a fixed noise floor and the current total output of all cells in the simulation discounted by its local path loss. Further, the signal to noise ratio (E_c/I_o) is calculated as the sum of the output of the pilot channel discounted by the path loss against the total noise at the location. Handover decisions, for each UE, are made based on smoothed simulated E_c/I_o measurements offset by the current CRE parameter, updated a fixed number of times each second.

The contributions 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.

III. EXPERIMENTS

The performance of the load balancing method is evaluated in several simulated scenarios, with varying number of cells and geographic UE densities. For each scenario we perform multiple simulation runs, where each run has a randomised initial condition with regard to cell and UE centre positions. Experiments are performed both with the load balancing mechanism engaged and disengaged. By quantifying and measuring the load balance we can then evaluate the impact of the balancing algorithm. The load balance, B , of the network is quantified simply as the sample variance of observed cell loads,

$$B = \frac{1}{n} \sum_{i=1}^n (l_i - \bar{l})^2, \quad (3)$$

where l_i is the actual individual cell loads, $\bar{l}$ is their sample mean, and n is the number of cells.

Throughout the scenarios tested, the method consistently manages to improve the balance in the network. We will have

a closer look at a specific, yet representative, scenario, where a 2.25 km^2 large area contains two macro nodes (20 W), three micro nodes (5 W) and four pico nodes (1 W) that together serve 1125 UEs. As described in section II, the UEs move according to a model based on Lévy walks, while generating network traffic patterns that are based on recorded traces.

Figure 3 depicts individual time series of load balance, each during 30 minutes, per run for 100 different simulation runs. Note that the initial rise from zero variance is a simulation artefact due to that the Bayesian estimators employed to measure the load require a few seconds worth of observations prior to providing accurate estimates. By comparing Figure 3(a) and (b) we see that the load balancing algorithm suppresses a large portion of high-variance loads. This result is summarised in Figure 4(a) by showing the average balance for all 100 experiments. In this specific scenario, the mean load variance is decreased with almost 50%. Further, as indicated by the error bars in Figure 4(a), the variance of balance over runs is also decreased when the proposed method is employed, implying more consistent and, in effect, more predictable load behaviour among cells.

Figure 4(b) confirms that the maximum cell load in the network decreases while the balancing mechanism is engaged, which is expected since the network is better at distributing loads between neighbouring cells (the initial drop is, again, due to the reason explained above). The decrease of maximum load is significant, comprising of about 10-15% of the non-balanced cell load. As is the case with load balance, the maximum load varies less between simulation runs when the balancing method is engaged, again resulting in more consistent and predictable network load behaviour.

IV. CONCLUSIONS AND PERSPECTIVE

We have outlined a novel load balancing solution for HetNets based on autonomous and distributed computation of local load targets that, as long as a majority of nodes calculates their target loads in the same way, will converge towards an ideal load distribution. Using these target loads we have also outlined a load redistribution mechanism based on scaling the difference between target and actual load to an available range of CRE values. The combination of these two mechanisms have been evaluated in a simulator incorporating UE traffic and mobility distributions, and a simple RAN simulator. Using the same target load calculation, other load redistribution mechanisms could also be used.

The main obstacle in attaining even greater load redistribution is the trade-off between the service quality for UEs left connected with a weak signal from a lightly loaded node, and connecting that UE to a node with a stronger signal but with a resulting load which will impact many user's service quality. For this reason alternative load redistribution objectives should be investigated, e.g. minimising the maximum risk of overload. Another aspect is the potential overhead in terms of additional hand-overs as the local load levels fluctuates. In our prototype, we have managed this by low-pass filtering the target load updates, but a more direct way of managing the trade-off

Fig. 3. Load balance (expressed as load variance over the nodes) for 100 simulator runs, (a) without and (b) with the load balancing mechanism engaged, for a scenario with 2 macro nodes, 3 micro nodes, 4 pico nodes and 1125 UEs in an area of 2.25 km^2 .

Fig. 4. Mean of (a) load variances and (b) maximum cell load over 100 simulator runs, with (solid line) and without (dashed line) the load balancing mechanism engaged, for the same scenario as described in Figure 3. Error bars show the standard error of the mean.

between hand-over overhead and dynamicity of the network may also be considered.

These two mechanisms (as well as others reported in [8]) are based on a library of generic software tools for distributed computation and Bayesian statistics that we aim to employ and expand for use in solving other RAN management problems.

Furthermore, the methods presented here are reactive, which means that they adapt to changes as they happen, such as a sudden increase of traffic demand. Another possible future direction is to develop techniques that instead are predictive, which is likely to be more efficient as demands are foreseen so that the system can adapt to them prior to their realisation. Such predictions could be made on the cell and even UE level, where, for instance, Bayesian estimators are maintained for predicting traffic behaviour.

ACKNOWLEDGMENT

The results presented here are based on research funded by Ericsson AB, Development Unit Radio (DURA).

REFERENCES

- [1] J. G. Andrews, "Seven ways that hetnets are a cellular paradigm shift," *IEEE Communications Magazine*, pp. 136–144, Mars 2013.
- [2] I. Siomina and D. Yuan, "Load balancing in heterogeneous lte: Range optimization via cell offset and load-coupling characterization," in *Communications (ICC), 2012 IEEE International Conference on*, June 2012, pp. 1357–1361.
- [3] H. Wang, L. Ding, P. Wu, Z. Pan, N. Liu, and X. You, "Dynamic load balancing and throughput optimization in 3gpp lte networks," in *Proceedings of the 6th International Wireless Communications and Mobile Computing Conference*, ser. IWCMC '10. New York, NY, USA: ACM, 2010, pp. 939–943. [Online]. Available: <http://doi.acm.org/10.1145/1815396.1815611>
- [4] P. Fotiadis, M. Polignano, D. Laselva, B. Vejlgard, P. Mogensen, R. Irmer, and N. Scully, "Multi-layer mobility load balancing in a heterogeneous lte network," in *Vehicular Technology Conference (VTC Fall), 2012 IEEE*, Sept 2012, pp. 1–5.
- [5] 3GPP Technical Specification Group: Radio Access Network, .
- [6] A. Lobinger, S. Stefanski, T. Jansen, and I. Balan, "Load balancing in downlink lte self-optimizing networks," in *Vehicular Technology Conference (VTC 2010-Spring), 2010 IEEE 71st*, May 2010, pp. 1–5.
- [7] Å. Arvidsson, D. Gillblad, and P. Kreuger, "Tracking user terminals in a mobile communication network," Patent PCT/EP2011/060090, June 2011, Ericsson AB. [Online]. Available: <http://patentscope.wipo.int/search/en/detail.jsf?docId=WO2012171574>
- [8] P. Kreuger, D. Gillblad, and A. Arvidsson, "zcap: A zero configuration adaptive paging and mobility management mechanism," *Journal of Network Management*, vol. 23, pp. 235–258, 2013.
- [9] K. Lee, S. Hong, S. J. Kim, I. Rhee, and S. Chong, "Slaw: A new mobility model for human walks," in *INFOCOM*. IEEE, April 2009, pp. 855 – 863.
- [10] C. Song, T. Koren, P. Wang, and A.-L. Barabasi, "Modelling the scaling properties of human mobility," *Nature Physics*, vol. 6, pp. 818–823, 2010.
- [11] C. Song, Z. Qu, N. Blumm, and A.-L. Barabasi, "Limits of predictability in human mobility," *Science*, vol. 327, no. 5968, pp. 1018–1021, 2010.
- [12] M. C. Gonzalez, A.-L. Barabasi, and C. A. Hidalgo, "Understanding individual human mobility patterns," *Nature*, vol. 453, pp. 779–782, June 2008.
- [13] V. Erceg, L. J. Greenstein, S. Y. Tjandra, S. R. Parkoff, A. Gupta, B. Kulic, A. A. Julius, and R. Bianchi, "An empirically based path loss model for wireless channels in suburban environments," *IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS*, vol. 17, no. 7, pp. 1205–1211, July 1999.