Identifying Cell Sector Clusters Using Massive Mobile Usage Records

Zhe Chen†
CommScope Inc., Singapore
zhe.chen@commscope.com

Emin Aksehirli†
Twitter Inc., Singapore
eaksehirli@twitter.com

***Abstract*—Optimizing capital expenditure (CapEx) has been an increasingly important objective in telco operators' cell planning process. Traditionally, neighbor cell relation is operationally managed and independent from capacity planning. In this paper, we present SCUT, an algorithm that uses massive mobile usage records to detect clusters of possible capacity-sharing sectors, such that capacity planning can be optimized based on coverage. SCUT analyzes shared usage to build a graph-based model of an operator's network and identifies its disjoint dense components as best-fit abstractions of clusters. Through analysis and benchmarking on real data, we demonstrate its scalability and potential to improve industry-standard site-based planning. SCUT has been deployed for a telco operator in Southeast Asia.**

***Index Terms*—CDR, Cell Sector Cluster, Shared Usage Graph, Clique Detection, Apache Spark**

I. Introduction

As 4G LTE becomes ubiquitous and 5G is on the horizon, cell planning has become increasingly important for telco operators. The ever-increasing demand for mobile broadband [1], [2] necessitates denser cell deployment for good customer experience. Hence, telco operators need optimal cell planning, which is considered crucial to their long-term success [2].

While telco operators are under pressure to continue investing in their infrastructure, they are also experiencing overall profit margin shrink [2]. Therefore, it is becoming more and more important for cell site upgrades and new site deployments to not only increase capacity but also improve Return on Investment of capital expenditure (CapEx).

The traditional planning process assesses the network at the cell site level. Neighbor cell relation is managed either manually using software tools [3], [4] in earlier technologies or automatically in LTE [5]. Independently, capacity planning forecasts the demand of each cell site and plans appropriate upgrades to handle the increased demand [4]. In some cases, the handover zone could enable a cell to handle a portion of the demand from a nearby cell during busy hours. Hence, upgrading one cell site or one sector could potentially provide enough coverage and capacity for the surrounding area, saving the need to upgrade more sites.

In this paper, we present SCUT (for Sector Clustering with User Transactions), a data-centric algorithm using users' mobile usage records to automatically identify sector clusters with capacity-sharing potential. The cluster information can be used in cell planning so that operators can selectively upgrade sectors and not over-upgrade nearby sectors at the same time, hence optimizing CapEx. We then study if clusters identified by SCUT can potentially improve the planning process. SCUT has been deployed for a mid-sized telco operator ("the Operator") in Southeast Asia, who is looking to embed CapEx optimization into their cell planning process, and its result is preliminarily validated by network planners from the Operator.

Contributions. This paper makes two main contributions. First, from the application perspective, we have designed, developed, and deployed SCUT. SCUT uses mainly *Call Detail Records* (CDR), or a derivation of it. Since CDR is used for billing purposes, it is one of the most well-maintained and available data sources from operators. Therefore, operators can directly employ our algorithm on their historical billing data without collecting special data sources, such as operational network data. Moreover, to handle a large volume of mobile usage records that telco operators typically collect, SCUT is highly scalable by design.

Second, within the SCUT algorithm, we utilize the concept of shared user-hours to quantify the effective usage of handover zones between pairs of cells or sectors through users' mobile usage. In our graph-based model, we introduced two ranking scores for cliques and use the rank to enumerate disjoint clusters in a directed weighted graph.

We evaluated SCUT using historical data from the Operator and proved that it can produce results using a reasonable amount of time and resources. We also used an independent dataset from the Operator to show the improvements of SCUT clusters over the industry standard.

II. Cell Planning

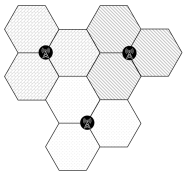
A. Background

Figure 1(a) illustrates a typical monopole *cell site* using directional antennas (*cells*). Each cell is identified in records by its Cell Global Identity (*CGI*). Figure 1(b) illustrates the theoretical wide-area coverage of multiple cell sites [6]. The coverage of each cell site, represented by a black dot, is split into multiple *sectors* (typically 3), represented by hexagons. Each cell covers a sector. Other deployment types, such as rooftop deployment, also employ sectorized coverage.

To provide service flexibility, 4G network operates on over 40 standardized frequency bands (*carriers*) [7]. Telco

† This work is completed while both authors were at DataSpark Pte. Ltd.

(a) Monopole Cell Site



(b) Sectorized Cell Site Coverage

Figure 1: Cell Planning Illustrations

operators typically install multiple cells operating on different carriers on the same sector.

The boundary of a sector is not clearly defined and handover zones are set up between sectors to allow the user equipment (UE), e.g. mobile phones, within it to switch from one cell to another for continued coverage while they are in motion. Handover can also occur when the currently-connected cell becomes congested and Mobility Load Balancing [8] directs the UE to reconnect to a less congested cell within range.

When a cell site is congested, users experience lower throughput and telco operators will take measures to increase its capacity. From a telco operator's perspective, the common measures are: adding sectors through electronic beam-splitting, adding carriers by installing new cells, and installing better antennas [6]. Among these measures, adding carriers requires knowledge of the existing cells in the sector, as the new cell needs to operate on a non-interfering carrier. Therefore, a sector is the smallest unit that allows upgrade planning.

B. Related Work

Early cell planning studies [9], [10] modeled signal strength of a cell antenna with its physical properties, such as transmission power, tilt angle, azimuth, etc. The interactions between cells, such as handover and interference are then defined based on the signal strength model. These studies may not be suitable for today's 4G network due to its vastly increased variety of frequency bands [7] and antenna types [6], as well as network heterogeneity [11].

More recent works took into account some of the 4G network complexity, such as Heterogeneous Network (HetNet) and Multiple-Input and Multiple-Output (MIMO), as surveyed in 2017 [11]. These works introduced new theoretical models for inter-cell interference, multi-carrier systems and more, while some have taken user distribution into consideration [12]. These models could still be challenged by the complexity of the actual deployment, and enriching with realistic operational data [13] can improve their relevance.

The total cost of ownership (TCO) is becoming a key objective in optimizing the cell planning process [14], [15], but variations in traffic demands, ignored by traditional cell planning approaches, can lead to capacity wastage and difficulty in interference management [15], resulting in higher TCO. Chew, Mo and Yeo [16] included mobility patterns in cell planning to optimize capacity. Wang and Ran [15] proposed a new cell planning framework to guarantee the quality of service, in which a given region is partitioned into planning zones with even traffic demand.

mobile usage records:
userID, date, hour, cgi, dataVolume, callDuration, smsCount

cell inventory:
cgi, carrier, lat, long, sectorID, siteID, locationType

Figure 2: Schema of Data Sources

In this paper, we aim to bridge the gap between cell-site-based planning and demand-based planning by providing clusters of sectors using user demand patterns mined from mobile usage records. This is similar to the concept of planning zone [15], and data in a similar format is also used to predict user activity level [17].

III. SCUT Overview

A. Data Sources

We use two types of data in SCUT: mobile usage records and cell inventory. Their schemata are shown in Figure 2 and are detailed in Appendix[1] B. The cell inventory contains operating carrier and other properties of each cell. The mobile usage records describe cells that each UE used in every hour and voice/SMS/data usage of the UE on the cell.

B. Algorithm Overview

We use the concept of shared user-hour to quantify the effective usage of handover zones between pairs of cells or sectors. A user-hour is a unit representing the mobile usage of a mobile subscriber within an hour. When multiple cells are used within a user-hour, handover has presumably occurred, and this user-hour is described as being shared among those cells. When more than two cells are used within a user-hour, we simplify the attribution of a shared user-hour to all unique pairs of cells from them. We plan to use data volume per cell per hour to calculate the percentage of shareable volume per cell/sector in a subsequent algorithm, hence we opted to exclude this factor from SCUT.

SCUT is designed to run in two successive stages, as illustrated in Figure 3.

Stage 1 attributes all observed user-hours in the records to construct a Shared Usage Graph (SUG), an undirected weighted graph $G(V, E)$. Each V is a cell and each $W(e), e \in E$ is the number of shared user-hours between the pair e. The SUG is then aggregated to sector level and normalized. This stage only runs once to generate SUG and Normalized SUG from a massive amount of usage records.

Stage 2 runs the Bron-Kerbosch algorithm on the Normalized SUG to enumerate maximal cliques. The cliques are ranked with two ranking scores and the highest-ranked set of disjoint cliques are selected as clusters. This stage has tunable parameters and may require multiple runs to get satisfactory clusters.

C. Why Bron-Kerbosch Algorithm?

There are numerous classes of algorithms that segment a graph based on different similarity metrics, such as distance, adjacency, and connectivity [18]. In our case, we define the

[1] Appendix is available at https://chen-zhe.github.io/files/2020-SCUT.pdf

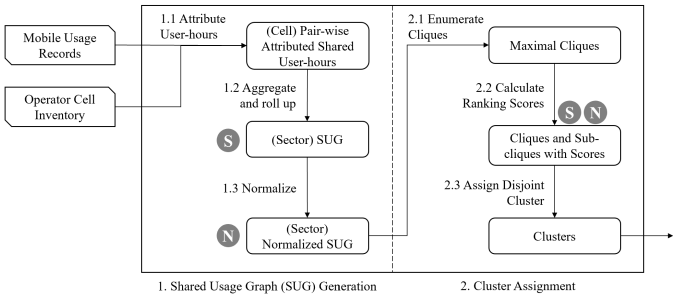
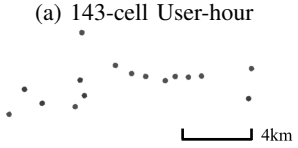


Figure 3: SCUT Data Flow Chart

(a) 143-cell User-hour



(b) 30-cell User-hour

Figure 4: Cell Locations of Outlying User-hours

clusters as a set of sectors serving a similar group of users. Hence, in the SUG (rolled up to sector level) generated with sufficient user-hour observations, we expect shared user-hours between every pair of sectors of a cluster. This means that a cluster must contain a subset of vertices that are all adjacent to each other in the SUG, which coincides with the definition of a clique in the graph theory. Therefore, maximal clique enumeration algorithms are the most suitable ones.

The SUG is a weighted undirected graph, however, the maximum weight clique algorithm [19] outputs a single maximum weight clique and does not enumerate cliques.

The Normalized SUG is a directed graph due to the normalization method used. Dense components [20] in a directed graph are analogous to cliques in an undirected graph. Greedy solutions such as the densest subgraph algorithm [21] solve the densest subgraph problem, but only outputs a single densest clique as well.

Hence, to enumerate all maximal cliques, we use the Bron-Kerbosch algorithm [22] by ignoring the edge weight and edge direction in a graph. Two ranking scores are introduced to select the disjoint cliques afterward, of which the Coherence Percentile is similar to the density definition [20] extended to a directed weighted graph.

IV. Methodology

A. Shared Usage Graph (SUG) Generation

Step 1: Shared User-hour Attribution (Algorithm 1). Each user-hour observed in mobile usage records gets a nominal value of 1, which is then attributed to every pair of cells recorded within the hour.

Table I: Coverage Range Estimates per Band

Frequency Band	**Urban Range** (km)	**Rural Range** (km)
LTE 700MHz	1.5	4.5
LTE 1800MHz	1.5	3
LTE 2600MHz	1.2	2

Algorithm 1: Shared User-hour Attribution

Input: $\mathcal{C}$ - set of connected cells of a user-hour
Output: Pairs of cells with attributed shared user-hours
for $(cell_i, cell_j) \leftarrow \mathcal{C}^2$ **do**
  if $distance(cell_i, cell_j) > range(cell_i) + range(cell_j)$ **then**
    return no attribution as user is moving
end
return $(cell_i, cell_j, 1.0/size(\mathcal{C}))$ for $\forall(cell_i, cell_j) \in \mathcal{C}^2$

However, a user could travel a great distance in an hour, as shown in Figure 4 (the underlying map is not shown). Attribution of such moving user-hour would not accurately describe the load-balancing handovers between cells. To filter them out, we obtained a set of estimated band ranges (table I) from the Operator's network engineering team and remove user-hours that cannot reach all used cells at a single place.

Step 2: SUG Aggregation. To generate the cell-to-cell SUG, the attributed user-hours between every observed pair of cells from the entire mobile usage records are summed up to become the edge attribute between vertices representing cells. The cell-to-cell SUG is then rolled up to a sector-to-sector SUG, but any shared user-hours between cells in the same sector would be removed.

Step 3: SUG Normalization (Algorithm 2). Global normalization methods, e.g. min-max scaling, would skew the result for vertices with low shared user-hours over all its edges. Our method normalizes edge attributes locally with regard only to its adjacent edges. It avoids the aforementioned issue but results in a directed graph, as each edge attribute gets normalized to two different values, that is, one per normalized edge direction. The normalized attribute of each edge is bounded between 0 and 1.

Time Complexity (detailed in Appendix D2). $O(r(1 + \frac{c}{u}))$ in the worst case, where r is the number of rows in mobile usage records, u is the number of users and c is the average number of cells that a user connects to within an hour. Thus, the runtime of this stage is linear to r and c.

Algorithm 2: SUG Normalization

Input: A - Adjacency matrix of SUG ($A_{ij} = A_{ji}$)
Output: A' - Adjacency matrix of Normalized SUG
$\overrightarrow{rowSum} \leftarrow A \cdot$ Column Vector $\vec{1}$
foreach $i \in$ *row index of* A **do**
  $A'_i \leftarrow A_i \div \overrightarrow{rowSum}_i$
end
return A'

Algorithm 3: Clique Enumeration

Input: G_N - Normalized SUG
Parameter: w - minimum weight threshold
Output: list of cliques as set of sectors
$G_F \leftarrow G(V \in G_N, \{E | E \in G_N, W(E) \geqslant w\})$
$cliqueList \leftarrow BronKerbosch(G_F)$
for $V_{Set} \leftarrow cliqueList$ **do**
  if $|V_{Set}| \geq 3$ **then**
    $subCliques \leftarrow \forall S \subset V_{Set}, |S| \geq 2$
    $cliqueList \leftarrow$ extend by $subCliques$
end
return $cliqueList$

Algorithm 4: Ranking Scores Calculation

Input: $cliqueList$ - list of cliques as set of sectors
Input: E_S - SUG edges. $W(E)$ retrieves shared user-hours
Input: E_{NS} - Normalized SUG edges. $W(E)$ retrieves normalized weight
Output: list of cliques tagged with *Total User-Hours* (TUH) and *Coherence Percentile* (CP)
for $V_{Set} \leftarrow cliqueList$ **do**
  $TUH \leftarrow \sum_{v_1, v_2 \in V_{Set}} W(E_S(v_1, v_2))$
  $CP \leftarrow \left\lfloor \sum_{v_1, v_2 \in V_{Set}} \frac{100 \cdot W(E_{NS}(v_1, v_2))}{|V_{Set}|} \right\rfloor$
  $V_{Set} \leftarrow$ tag with TUH and CP
end
return $cliqueList$

B. Cluster Assignment

Step 1: Clique Enumeration (Algorithm 3) is further illustrated in Appendix C. The Normalized SUG has its edges lower than minimum weight threshold (*w*) removed. It is then treated as an undirected graph by ignoring edge weights and directions. If a pair of vertices have one or more edges, the undirected graph will have one edge. Bron-Kerbosch algorithm is then applied to get a list of maximal cliques.

Sub-cliques of maximal cliques with at least three vertices are also enumerated. As such, if one vertex of a 3-clique gets assigned to another cluster due to higher ranking, the remaining 2-clique may still be assigned as a cluster.

Step 2: Ranking Scores Calculation (Algorithm 4). We introduce two ranking scores for the cliques: *Coherence Percentile* (CP) and *Total User-Hours* (TUH).

CP is the integer percentile of the average percentage of shared user-hours of a sector to a particular clique. A clique is considered more coherent if more shared user-hours (handovers) is observed within clique members rather than between clique members and non-clique members. This score is agnostic to clique size and is bounded between 0 and 100.

TUH is the sum of shared user-hours attributed to all edges within a clique. This factor generally favors larger cliques and a bigger value is desired, as a bigger cluster is of more interest than a small cluster. TUH is unbounded.

Step 3: Disjoint Cluster Assignment (Algorithm 5). After filtering out cliques without sufficient Coherence (*minCP*) and TUH (*minTUH*), all cliques are ranked in descending order by first CP and then TUH. Traversing through each clique in sorted order, a clique that is disjoint with previously-selected clusters is selected as a cluster. Sectors without any cluster assignment will become single-sector clusters.

Algorithm 5: Disjoint Cluster Assignment

Input: $taggedCliques$ - list of tagged cliques
Parameter: $minCP, minTUH$
Output: Cluster Assignment
$visitedSectors \leftarrow \emptyset$
$assignedClusters \leftarrow \emptyset$
filter $taggedCliques$ by $minCP$ and $minTUH$
sort $taggedCliques$ by CP and then TUH
for $V_{Set} \leftarrow taggedCliques$ **do**
  if $visitedSectors \cap V_{Set} = \emptyset$ **then**
    $visitedSectors \leftarrow visitedSectors \cup V_{Set}$
    $assignedClusters \leftarrow$ add V_{Set}
end
return $assignedClusters$

Time Complexity (detailed in Appendix D3). $O(a(m^2 + \log a + 1))$ in the worst case, where $a = 3^{s/3}2^m$, s is the number of sectors on the network and m is the average clique size. In our experiments, the SUG is sparsely connected with mostly 2-cliques and 3-cliques, thus this stage is still reasonably fast.

V. Experiments

A. Experiment Setup and Data Sources

To study how the three parameters described in section IV-B impact the clustering stability of SCUT over time and to benchmark its performance against the site-based planning approach, we ran a grid search on SCUT with multiple combinations of the parameters and validated results with an independent dataset.

We obtained one month (30 days) of anonymized nationwide mobile usage records and cell inventory from the Operator as inputs. We also obtained cell performance data (schema shown in Appendix B) from the Operator's Operation Support System (OSS) for the same month.

The mobile usage records have 13 billion rows, from which 5.5 billion user-hours are observed. Figure 5 shows the distribution of the number of connected cells of every user-hour. Since CDR only captures active usage, users' network idle time, e.g. during sleeping or surfing with WLAN, is not represented. 46.1% of the user-hours connects to a single cell, and 99.9% of the user-hours connects to less than 30 cells (we used this value as the threshold of moving user-hours).

We selected two outlying user-hour examples and plotted locations of the observed cells onto a map in Figure 4. As the cells in both user-hours are close to major roads, we can conclude that these users are moving within that hour. Our filter described in section IV-A effectively removed them while keeping 92.7% of multi-cell user-hours.

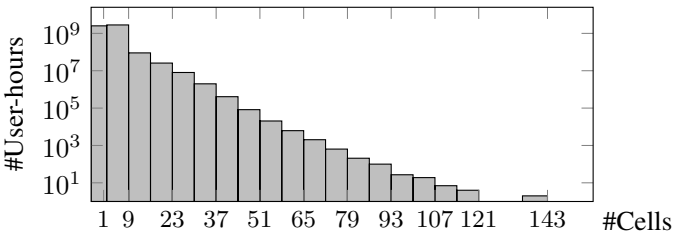


Figure 5: Histogram of Connected Cells per User-hour

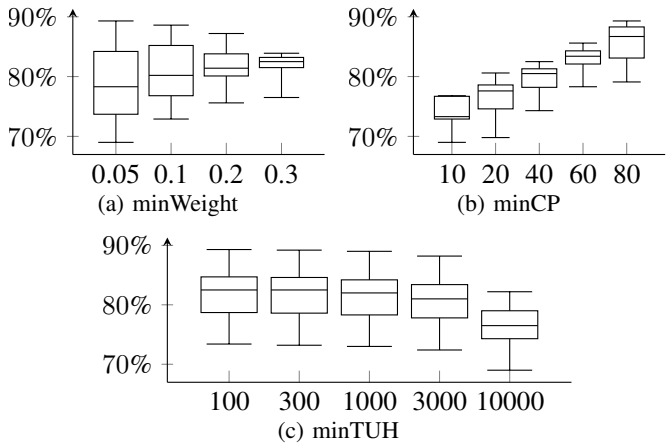


Figure 6: Exact Match Percentage Distribution

The cell inventory shows that the Operator owns around 200,000 cells across the country, installed on more than 8,000 cell sites. Each site contains 3 to 6 sectors. We choose to only include outdoor cells for clustering since indoor cells have a limited range and a different site definition.

The cell performance data has the hourly transmitted data volume from each cell.

In terms of performance, SCUT is implemented in Scala on Apache Spark and is tested on a Hadoop cluster (CPU: Intel Xeon E5-2640 v3, OS: Red Hat Enterprise Linux 6.9) using 106 cores and 421 GB RAM in total. The SUG Generation stage generated the SUG and Normalized SUG using all records in 49 minutes. The Cluster Assignment stage generated a list of hard clusters in two minutes.

B. Stability Analysis

We partitioned 30 days of data into three disjoint segments of 10 consecutive days and ran SCUT with each segment using all combinations of the parameters shown in Figure 6. We then calculated percentage of the same cluster being detected from all three segments as $|R_1 \cap R_2 \cap R_3| \div \max(|R_1|, |R_2|, |R_3|)$, where R_n is the set of assigned clusters using input from segment n. Figure 6 shows the variations of percentage when one parameter is fixed.

This analysis shows that with proper parameter settings, SCUT is able to achieve over 80% exact matches between the segments and over 90% when partial matches are included. Figure 6(b) also shows that increasing *minCP* significantly improves the stability of the result. This is consistent with the design of the CP score.

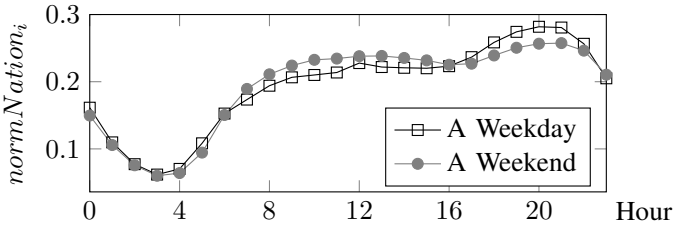


Figure 7: Normalized Nationwide Daily Patterns

C. Benchmarking Against Industry Practice

If sectors on the same cluster serve a more similar group of users as compared to sectors on the same cell site, we may conclude that clusters are more consistent for planning. To benchmark, SCUT ran on the full month of data, so the observed shared user-hours roughly triples as compared to section V-B and the maximum $minTUH$ tested in this section are tripled as compared to Figure 6(c).

We assume that when cells serve a similar group of users, they exhibit similar daily usage patterns. We use the time series of the hourly data volume of each cell per day (from OSS data) to represent its usage pattern of the day. Each aggregation unit, i.e. sector, site or cluster, uses the aggregated data volume from all cells within to represent its usage pattern. We use Cosine Similarity (Appendix A) to quantify the similarities between two time series $A_{1..24}$ and $B_{1..24}$.

All cells exhibit a similar usage pattern due to human diurnality, as shown in Figure 7 by summing up the data volume from all cells in every hour of a day. To mitigate any bias introduced by such patterns, we adopt the concept from Cosine Similarity to normalize each time series using equation 1 and use the residual pattern $residualA_i = normA_i - normNation_i$(daily) of each cell for comparison instead.

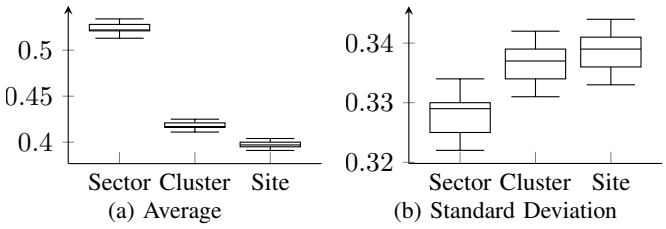
$$normA_i = \frac{A_i}{\sqrt{\sum_{i=1}^{n} A_i^2}} \tag{1}$$

For clusters produced by each set of parameters, we calculate similarities between the pattern of each cell and that of its sector, site, and cluster, and then characterize the overall similarity between cells and each of the aggregation units by its mean $\bar{x}$ and standard deviation s. Only multi-sector clusters and sites containing cells from these clusters are included in the similarity comparison.

The result shows that overall, cells in the same sector behave 33.6% more similarly ($\bar{x} \approx 0.524$) as compared to site ($\bar{x} \approx 0.392$). This is expected as cells in the same sector face the same direction, and therefore cover a similar area and users. A cell site contains different sectors, hence covers a wider variety of users. This also proves that our previous assumption is accurate, to a certain extent.

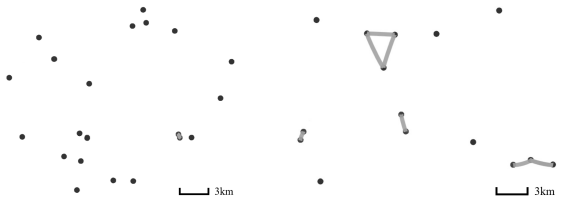
Equation 2 scores the result of SCUT with a set of parameters. It jointly evaluates the improvement in the similarity of clusters over that of cell sites, and the closeness of the similarity of clusters to that of sectors, which are shown to be better.

$$score = \frac{\bar{x}_{cluster}}{\bar{x}_{sector}} - \frac{\bar{x}_{site}}{\bar{x}_{cluster}} + \frac{s_{sector}}{s_{cluster}} - \frac{s_{cluster}}{s_{site}} \tag{2}$$



(a) Average (b) Standard Deviation

Figure 8: Best Case Residual Similarity



(a) Urban Area (b) Rural Area

Figure 9: Cluster Visualizations

Figure 8 summarizes the residual similarity of the best five sets of parameters. Using the best one (*minWeight*=0.3, *minCP=40*, *minTUH*=30000), SCUT clusters are able to obtain 5.1% higher similarity as compared to sites, while sectors obtain 31.2% higher similarity. Sectors within a cluster may only partially serve the same group of users, so the minor improvement still shows that user overlap is greater for sectors in a cluster as compared to those in a cell site.

With the best set of parameters, we assigned 1465 multi-sector clusters, of which 485 are 3-sector clusters and 980 are 2-sector clusters, partially illustrated in Figure 9. 25.9% of the clusters detected are equivalent to the operator-defined cell sites, confirming that in some cases, a site is still a good representation of a cluster of sectors. As this parameter set gives the most confident clusters, only 13.8% of sectors owned by the Operator are clustered, and an operator can use them to aid their traditional planning process. Urban areas are more densely covered, so defining tightly-linked clusters could require a different set of parameters as compared to rural areas. Rural areas appear to be more optimized using this set of parameters, as more clusters span across different cell sites in rural areas.

VI. Conclusion

Driven by a telco operator's need to optimize cell planning for CapEx, we developed SCUT to identify sector clusters with capacity-sharing potential using available data sources at a massive scale. It is fully data-driven and models a network from the perspective of users. It identifies sector clusters for demand-based planning. Since sectors in a cluster potentially share user load, operators can avoid over-upgrades, leading to more optimize CapEx. We learned from experiments that sector-based demand analysis is the most consistent, but clusters can potentially reduce capacity wastage during cell planning and are still more consistent than industry-standard planning with cell sites. The algorithm has been deployed in the Operator's environment and has received some validation.

In our future work, we plan to get more relevant data sources to measure the performance of SCUT. We are also keen to explore other use cases of the SUG using graph algorithms.

References

[1] J. Rodriguez, *Fundamentals of 5G Mobile Networks*. John Wiley & Sons, Jun 2015.

[2] F. Dickgreber, C. Campanini, S. Grabowski, and T. Sorensen. (2015, Feb) The Future of Telecom Operators in Europe. [Online]. Available: https://www.kearney.com/communications-media-technology/article?/a/the-future-of-telecom-operators-in-europe

[3] Groundhog CovMo Features. [Online]. Available: https://www.ghtinc.com/covmo/

[4] TEOCO ASSET Planning Suite Features. [Online]. Available: https://www.teoco.com/products/planning-optimization/

[5] A. Dahlen, A. Johansson, F. Gunnarsson, J. Moe, T. Rimhagen, and H. Kallin, "Evaluations of LTE Automatic Neighbor Relations," in *2011 IEEE 73rd Vehicular Technology Conference (VTC Spring)*, May 2011, pp. 1–5.

[6] "MIMO and Smart Antennas for 3G and 4G Wireless Systems - Practical Aspects and Deployment Considerations," 3G Americas, Bellevue, WA, USA, White paper, May 2010.

[7] M. Mueck, D. C. Karls, Ingolf, and B. Badic, *Rolling Out 5G: Use Cases, Applications, and Technology Solutions*. Apress, Jun 2016.

[8] (2010, Mar) 3GPP Acronyms: Self-Organizing Networks. [Online]. Available: https://www.3gpp.org/technologies/keywords-acronyms/105-son

[9] S. Hurley, "Planning effective cellular mobile radio networks," *IEEE Transactions on Vehicular Technology*, vol. 51, no. 2, pp. 243–253, March 2002.

[10] D. Amzallag, M. Livschitz, J. Naor, and D. Raz, "Cell Planning of 4G Cellular Networks: Algorithmic Techniques and Results," in *2005 6th IEE International Conference on 3G and Beyond*, Nov 2005, pp. 1–5.

[11] A. Taufique, M. Jaber, A. Imran, Z. Dawy, and E. Yacoub, "Planning Wireless Cellular Networks of Future: Outlook, Challenges and Opportunities," *IEEE Access*, vol. 5, pp. 4821–4845, 2017.

[12] E. Yaacoub and Z. Dawy, "LTE radio network planning with HetNets: BS placement optimization using simulated annealing," in *MELECON 2014 - 2014 17th IEEE Mediterranean Electrotechnical Conference*, April 2014, pp. 327–333.

[13] X. Xu, I. Broustis, Z. Ge, R. Govindan, A. Mahimkar, N. K. Shankaranarayanan, and J. Wang, "Magus: Minimizing Cellular Service Disruption During Network Upgrades," in *Proceedings of the 11th ACM Conference on Emerging Networking Experiments and Technologies*, ser. CoNEXT '15. New York, NY, USA: ACM, 2015, pp. 21:1–21:13.

[14] R. K. Taplin, D. M. Ryan, S. M. Allen, S. Hurley, and N. J. Thomas, "Algorithms for the Automatic Design of WiMAX networks," in *The 2007 International Conference on Next Generation Mobile Applications, Services and Technologies (NGMAST 2007)*, Sep. 2007, pp. 322–327.

[15] S. Wang and C. Ran, "Rethinking cellular network planning and optimization," *IEEE Wireless Communications*, vol. 23, no. 2, pp. 118–125, April 2016.

[16] Y. H. Chew, R. Mo, and B. S. Yeo, "Inclusion of Mobility in Cell Planning: Multi-Period Joint Optimization," in *2009 IEEE 70th Vehicular Technology Conference Fall*, Sep. 2009, pp. 1–4.

[17] C. Luo, J. Zeng, M. Yuan, W. Dai, and Q. Yang, "Telco User Activity Level Prediction with Massive Mobile Broadband Data," *ACM Trans. Intell. Syst. Technol.*, vol. 7, no. 4, pp. 63:1–63:30, May 2016.

[18] S. E. Schaeffer, "Survey: Graph Clustering," *Comput. Sci. Rev.*, vol. 1, no. 1, pp. 27–64, Aug. 2007.

[19] H. Jiang, C.-M. Li, and F. Manya, "An Exact Algorithm for the Maximum Weight Clique Problem in Large Graphs," in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, 02 2017.

[20] R. Kannan and V. Vinay, "Analyzing the structure of large graphs," Jul 1999, retrieved from http://www.cs.yale.edu/homes/kannan/Papers/webgraph.pdf.

[21] S. Khuller and B. Saha, "On Finding Dense Subgraphs," in *Automata, Languages and Programming*, S. Albers, A. Marchetti-Spaccamela, Y. Matias, S. Nikoletseas, and W. Thomas, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 597–608.

[22] C. Bron and J. Kerbosch, "Algorithm 457: Finding All Cliques of an Undirected Graph," *Commun. ACM*, vol. 16, no. 9, pp. 575–577, Sep. 1973.

APPENDIX

A. Cosine Similarity

Similarity between two n-dimension vectors $A_{1..n}$ and $B_{1..n}$ is defined as:

$$\frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \tag{3}$$

B. Detailed Data Schemas

Mobile Usage Records	
Field Name	***Data Type***
userID	String
date	String
hour	Integer
cgi	String
dataVolume	Double
callDuration	Double
smsCount	Long
Cell Inventory	
Field Name	***Data Type***
cgi	String
frequencyBand	String
lat	Double
long	Double
sectorID	String
siteID	String
locationType	String
Cell Performance	
Field Name	***Data Type***
cgi	String
date	String
hour	Integer
dataVolume	Double

C. Shared Usage Graph and Clique Enumeration

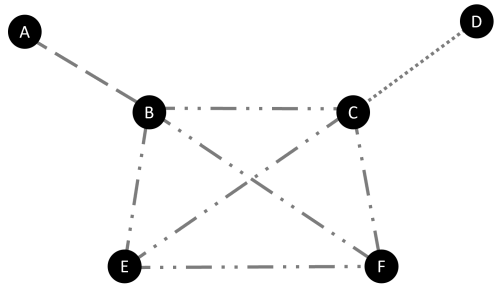


Figure 10: Shared Usage Graph Illustration

Figure 10 is a shared usage graph. Each vertex is a sector and each edge is tagged with the shared user-hours between two sectors. Different line types show different maximal cliques in the graph.

Maximal cliques: {A, B}, {C, D}, {B, C, E, F}

Sub-cliques of {B, C, E, F}: {C, E, F}, {B, E, F}, {B, C, F}, {B, C, E}, {B, C}, {B, E}, {B, F}, {C, E}, {C, F}, {E, F}

D. Time Complexity Derivation

1) Assumptions and Notations: Assuming the following components are used in implementing the algorithm:

- Hash tables with amortized $O(1)$ access time
- Sorting algorithms with $O(n \log n)$ run time

Denoting the quantities of the following SCUT algorithm inputs:

- r: number of rows in mobile usage records
- u: number of users
- c: average number of cells that a stationary user connects to within an hour
- s: number of sectors on the network
- m: average clique size (number of vertices in the clique)

2) Share Usage Graph Generation: In step 1 (Shared User-hour Attribution), a full table scan is first performed on mobile usage records to collect connected cells per user per hour. By using a hash table with each unique user-hour as a key, this would take $rO(1) = O(r)$ operations to complete. This step would produce an intermediate result of $\frac{r}{uc}$ rows describing every user-hour. For each user-hour, filtering of moving ones and nominal value attribution run on every pair of cells, i.e., ${}_cC_2 = \frac{c(c-1)}{2}$ pairs within each user-hour. Thus, another $\frac{r}{uc}\frac{c(c-1)}{2} = \frac{r(c-1)}{2u} \equiv O(\frac{rc}{u})$ operations are needed and this generates an intermediate result of $O(\frac{rc}{u})$ pairs.

In step 2 (SUG Aggregation), a full table scan is performed on all pairs of cells discovered in step 1. Using a hash table to record sector-pair-wise shared user-hours, $O(\frac{rc}{u})$ operations are required to generate the sector-to-sector SUG as a list of adjacent vertices.

In step 3 (SUG Normalization), a hash table is first used to accumulate total shared user-hours per sector. This can be done co-currently with step 2. In the worst case, each cell is in a unique sector, thus the sector-to-sector SUG is the same as the cell-to-cell SUG. To normalize the SUG, $O(\frac{rc}{u})$ operations are needed to scan through the SUG.

In total, the time complexity of the Shared Usage Graph Generation stage is $O(r) + 3O(\frac{rc}{u}) \equiv O(r(1 + \frac{c}{u}))$.

3) Cluster Assignment: In step 1 (Clique Enumeration), an optimized version of the Bron-Kerbosch algorithm is used, which has a time complexity of $O(3^{s/3})$ to enumerate at most $3^{s/3}$ maximal cliques (*On Cliques in Graphs*, J. W. Moon and L. Moser, 1965), where s is the number of graph vertices (number of sectors in our case). Each m-clique has $\sum_{i=2}^{m-1} {}_mC_i = \sum_{i=0}^{m} {}_mC_i - {}_mC_0 - {}_mC_1 - {}_mC_m = 2^m - m - 2$ sub-cliques. Hence, there would be at most $3^{s/3}(2^m - m - 1)$ cliques and sub-cliques and enumerating all cliques takes $O(3^{s/3}2^m)$ operations.

In step 2 (Ranking Scores Calculation), we implemented the SUG and NSUG lookup using the sector-pair-wise hash tables on each pair of vertices within every clique. A m-clique has ${}_mC_2$ pairs of vertices. Thus, this step takes $O(3^{s/3}2^m)\frac{m(m-1)}{2} \equiv O(3^{s/3}2^m m^2)$ operations.

In step 3 (Disjoint Clique Assignment), the cliques are sorted with a good sorting algorithm in $O(a \log a)$ time and

then looped through in $O(a)$ time, where $a = 3^{s/3}2^m$. Thus, this step takes $O(a \log a)$ time.

In total, the Cluster Assignment stage takes $O(a) + O(m^2 a) + O(a \log a) \equiv O(a(m^2 + \log a + 1))$, where $a = 3^{s/3}2^m$.