Attwood, A, Lamb, DJ and Abuelmaatti, O

Position-relative identities in the internet of things: An evolutionary GHT approach

Position-relative identities in the Internet of Things; an evolutionary GHT Approach

A. Attwood, D. J. Lamb, O. Abuelmaatti

Abstract— The Internet of Things (IoT) will result in the deployment of many billions of wireless embedded systems creating interactive pervasive environments. It is envisaged that devices will cooperate to provide greater system knowledge than the sum of its parts. In an emergency situation, the flow of data across the Internet of Things may be disrupted, giving rise to a requirement for machine-to-machine interaction within the remaining ubiquitous environment. Geographic Hash Tables (GHTs) provide an efficient mechanism to support fault-tolerant rendezvous communication between devices. However, current approaches either rely on devices being equipped with a GPS or being manually assigned an identity. This is unrealistic when the majority of these systems will be located inside buildings and will be too numerous to expect manual configuration. Additionally when using GHT as a distributed data store, imbalance in the topology can lead to storage and routing overhead. This causes unfair work load, exhausting limited power supplies as well as causing poor data redundancy. To deal with these issues we propose an approach that balances graph-based layout identity assignment, through the application of multi fitness genetic algorithms. Our experiments show through simulation that our multi fitness evolution technique improves on the initial graph-based layout, providing devices with improved balance and reachability metrics.

Index Terms— data centric storage, evolutionary computing and genetic algorithms, information dispersal, load balancing, Wireless sensor networks

I. Introduction

IDENTITY and addressing schemes are a key requirement of any network system. Such a scheme enables a device to be contacted and for the remote device to respond. In wireless mesh networks, devices are not simply endpoints but also route data; enabling the forwarding of information between endpoints. In the context of fixed wired networks, the routing function is commonly performed by dedicated devices; however, in wireless networks, devices must cooperate to enable global reachability. When all nodes in the network are able to communicate with one another, the network is said to have converged. In order to facilitate convergence in a wireless multi-hop network, devices must run a routing process that provides a mechanism for each device to construct a local routing data structure. These local data structures enable a device to make forwarding decisions that result in a message being passed closer to its intended destination. Each forwarding decision is determined by a routing algorithm that operates on the data structure held by the individual device. In wireless multi-hop networks, information is forwarded from device to device until it reaches its final destination.

Any routing and non-broadcast network architecture relies on parties being able to identify the originator and destination, in addition to suitable intermediate forwarding points in the network. For efficient or power-constrained routing, each device must have sufficient knowledge of the topology to forward the packets appropriately communicating nodes.

In an IoT network, devices will be assumed to have IP (Internet Protocol)-compliant identities. Initially, it was considered that the application of IP on constrained devices was unrealistic; these devices may need to run for years on a set of batteries, and as such have limited processing, storage and bandwidth capabilities. However, there have been an increasing number of implementations of IPv6 stacks targeting low-power devices. 6LowPAN is one such low-power variant of IPv6; utilising header compression to enable the transmission of IPv6 within the limited 802.15.4 link layer frame [1]. Moreover, 6LowPAN, owing to the massive IPv6 address space, provides the opportunity for every IoT device to have a globally-unique identifier. Coupling this identifier with the UDP-bound Constrained Application Protocol (CoAP) [2] service, being developed by the IETF's CoRE working group, provides a full service URI; e.g. `coap://fe80::202:b38e:ac13/pressure`. Routing 6LowPAN packets between nodes is often accomplished using IPv6 routing protocol for low-power, lossy links (RPL) [3].

Wireless Internet of Things devices will be deployed within private homes, industrial buildings and public spaces. In normal operating conditions, IoT nodes may co-operate with each other and a central gateway to connect them to a global identity space – usually the Internet. However, if devices are unable to communicate with each other or their gateway, then significant portions of the network risk losing the ability to transmit or relay information. When deployed in safety or mission-critical settings, devices must be able to operate to some degree during a systems failure. Furthermore, in many deployment situations, it may be desirable that remaining

Andrew Attwood is with the School of Computing, Mathematics and Digital Technology, Manchester Metropolitan University, Manchester, M1 5GD. E-mail: A.Attwood@mmu.ac.uk.

David Lamb is with the School of Computing and Mathematical Sciences, Liverpool John Moores University, Liverpool, L3 3AF. E-mail: D.J.Lamb@ljmu.ac.uk.

Omar Abuelmatti is with the School of Computing and Mathematical Sciences, Liverpool John Moores University, Liverpool, L3 3AF. E-mail:

O.E.Abuelmaatti@ljmu.ac.uk.

active devices can provide some insight into the last known state of the failed or isolated devices.

Distributed Hash Tables (DHTs) provide a mechanism to store information in a distributed manner; holding redundant copies of data at selected nodes about the network. Geographical Hash Tables (GHTs) modify this approach to provide nodes with both an identity and mechanism to route information within a wireless mesh network [4]. However, both of these approaches suffer shortcomings concerning reachability and distribution effectiveness, when applied in real-world deployment arrangements.

In response, this work proposes EBL-GHT (Evolve, Balance, Localise – Geographic Hash Tables); a mechanism to provide position-relative virtual identities that preserve or improve reachability and provide balanced key allocation for nodes in position relative topologies. It is intended that once deployed the identities would be used by a routing protocol such as Greedy Perimeter Stateless Routing (GPSR).

The rest of this paper is organised as follows: Section II provides a discussion of issues relating to Distributed and Geographical Hash Tables, while Section III details our background research. Section IV provides an overview of our proposed localisation mechanism. Section V provides an analysis of our approach. Section VI provides a summary and identifies the direction for future work.

II. Hash Tables: Keying and Address Spaces

When allocating nodes with an identity within the available identity space, it is important that the share of the address space that maps to that node is proportional. If the address space is imbalanced, then in a failure situation, a disproportionate loss of data and connectivity may occur. However, GHTs, when underpinned by a uniform hash function, can result in an unbalanced address space. To briefly illustrate this point, Fig. 1 shows a simple example of a one-dimensional address space. In this example, due to the physical placement of the nodes, device 20 is assigned a disproportionately-large portion of the address space.

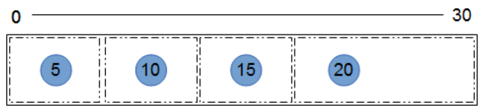

Fig. 1. Address Space Imbalance – This simplified diagram depicts the imbalance caused in GHT where a node is allocated an identity that positions it within a disproportionally large region of address space.

This work investigates this balancing problem inherent to the use of GHT address spaces in wireless Internet of Things sensor networks. In GHTs, a node's identity relates to its physical position—either to a common reference, such as GPS, or simply relative to its position in regard to its neighbours [4]. Relating a node's position to its identity in the hash table can provide additional resilience when storing information, owing to the fact that distance in the virtual address space also results in the physical separation of data within the sensor network. Distributing information between a set of autonomous peers provides a number of benefits over pre-planned orchestrated data distribution schemes. One important property is the even distribution of data between the collaborating nodes. If data is not distributed evenly, there is a risk of overloading individual nodes storage and processing capabilities. The risk of data loss due to node failure also increases.

DHTs are usually constructed above existing converged identity spaces. On the Internet, devices have an established identifier, commonly an IP address, allocated to the device by an upstream service provider. Devices on the Internet can use this identifier to achieve global reachability, providing the opportunity for connected nodes to construct an overlay DHT. Nodes wishing to join a DHT generate a random identity and contact a node that is already a member of the DHT to initiate the joining process. The discovery of a DHT and the joining process is usually referred to as 'bootstrapping'. The process of nodes joining with random identifiers should provide an even distribution of identities in the key space, resulting in nodes taking responsibility for an even portion of the identifier space.

The key is usually expressed as an integer of a predefined bit length. As such, the available address space is a product of this integer size. It is important that there is sufficient space so that nodes can pick an identity at random with a low probability of collision. It is also important that the keying space is sufficiently large so that data elements that are keyed into the space do not collide. Upon the node joining the DHT overlay, devices create neighbour relationships with other nodes in the overlay; the only nodes a device will contact directly using the underlying protocol, e.g. TCP/IP.

Messages that require routing are passed between neighbours of the overlay using the distance of the key to the neighbour entry as a method of forwarding data closer to its destination. This process can introduce path stretch [5] as nodes bypass the topological view of the supporting network and instead forward information based on the overlay topology. It is this property that makes overlay-based distributed hash tables unsuitable for sensor network deployments; however, there is ongoing work to make adaptations to improve the use of overlay protocols in wireless environments [6].

As an alternative to the traditional overlay-based DHT found on the Internet, wireless sensor networks can utilise position-relative identity spaces. In such schemes, nodes are provided with an identity relative to their physical location and/or another relative position metric, e.g. hop distance. A device can use its own identifier and those allocated to its neighbours to route information closer to a destination node—a process referred to as 'greedy forwarding'. Devices can use the same mechanism to distribute information into the GHT, as is found in a regular DHT; nonetheless, when using GHT, there is no underlying service that can be used to directly identify a device; rather, this information must be stored in the overlay. This is commonly referred to as a distributed 'indirection point; requiring communicating devices to first lookup the current position-relative identity using the hash of the known identity. This relies on the node with which you want to communicate, having already performed the same hash function on its own identity and stored this in the overlay with its current address as the data element.

Before a device can take part in a GHT, it must first be associated with an identity that falls within the addressable range of the network. The network must be bounded; in a GHT, this is usually the x and y coordinates between 0 and N, where N is dependent on the key length. If the devices in the network are keying data using a consistent hash function, such as MD5 on unique data, this should see the even distribution of keys in the coordinate space. If there is an even distribution of nodes in the coordinate space—either through physical location or virtual coordinates—then there is a good probability that each node will have an equal proportion of the keys to be stored. However, geographic hash functions demonstrate undesirable behaviour with regard to even distribution of keys, explored below.

For nodes to communicate, they need to be aware of their position so they can assume an appropriate identity, enabling them to forward and receive information. Commonly, schemes point to the use of GPS [7], although that has complications due to cost and the nature of deployment. In the absence of a reliable mechanism to determine location from fixed/physical external reference points, nodes can use a co-operative localisation approach.

Individual sensors can detect their immediate neighbours in the network. Using this per-node neighbour information, localisation algorithms attempt to recreate the topology, identifying suitable candidate coordinates to be allocated to the individual nodes. Both localisation and GPS-based position schemes provide the GHT with an approximated location of the device in the physical space. If the nodes are physically spread evenly, then storage load will accordingly be distributed evenly. If they are not, the network will suffer from imbalance; individual nodes will be the closest available identity for a large portion of the address space. In an effort to preserve the robustness of the distribution, the scheme proposed should retain the distance assurance properties of the GHT with the balancing properties of a standard DHT.

III. Position-Relative Distributed Hash Tables

A. Distributed and Geographical Hash Tables

DHT provides a useful abstraction to facilitate reliable and robust data dissemination in wireless sensor networks. There are three main approaches to building a DHT in wireless sensor networks: 1) Overlay DHT, where the address space is built on top of an existing converged protocol [6]; 2) Virtual, where nodes construct a tree [8]; and 3) physical location-based schemes [4], which map N-dimensional spaces onto the actual or estimated physical location of the node.

In [5], Awad et al. propose a virtual location scheme, referred to as Virtual Chord Protocol (VCP). In the scheme, nodes are provided with an identity within the range spanning 0–1. As nodes connect, they obtain an identity relating to that of a neighbour; this identity is used to route packets, as well as identity keys, that map to that node. Such a scheme may create unbalanced address spaces. This is intrinsic to their address allocation mechanism; as nodes join the system, the existing address space at a particular location will be partitioned. Depending on the ordering of nodes joining the system, areas of the address space may have been heavily partitioned and areas may have little partitioning; this results in data items being disproportionately placed at areas of low partitioning, resulting in an imbalance.

Geographic Hash Tables, as proposed by Ratnasamy et al. in [9], detail the Data Centric Storage where user data is pushed into an identity space formed by the physical real-world coordinates obtained via GPS sensors. This type of scheme can be useful when the accuracy of placement is essential; however, the strictness of identity related to a physical real-world position relies on the good physical distribution of nodes throughout the coordinate space to combat imbalance. Alternative similar approaches may use localisation in an effort to estimate the position of the nodes. This does nothing to improve the balancing issue, but does remove the requirement for GPS sensors. Alabno et al in [10] address the issue of non-uniformity of data placement. The approach outlined makes an estimation of network density dividing the address space into areas with coordinating nodes, the scheme relies on individual nodes knowing their geographic position within the network.

Scatterpastry [6] can be implemented using either overlay or underlay DHT. When using the overlay mechanism, there must be an existing Layer 2 frame-forwarding mechanism in place on the network, such as Destination-Sequenced Distance Vector (DSDV), for example[11]. However, the issue with this sort of approach is that the transmission in the DHT space causes path stretch in the Layer 2 space.

B. Localisation

Localisation algorithms provide a mechanism to identify the positions of individual nodes within wireless networks. Such information can then be provided to the individual devices so they can make forwarding decisions. Depending on the network deployment requirements, it might be possible to equip a subset of the nodes with a GPS or physically record their location. This provides valuable information when attempting to identify the location of the remaining nodes. These extrapolative, absolute location approaches are referred to as anchor-based localisation schemes. In [12], nodes use the hop-based position estimate from GPS-enabled device, which leads to a decentralised system, although it is reliant on the external GPS system and the continued operation of the subset of devices equipped with GPS receivers.

Anchor-free localisation does not require information external to the sensor network; instead, it utilises the information from the sensor network with the aim of estimating the relative positioning of devices. Usually, individual devices transmit the information they hold about the network back to a central co-ordinating unit. The information passed could include the following: Neighbour Identities, Node Identity, Signal Strength or incoming packets (RSSI), bit error rates, and ultrasonic/temperature or other sensory information. This information can be used by the localisation algorithm with the aim of determining the location of the devices.

Owing to the high search space, it is common for approaches to use probabilistic meta-heuristics. For example, in [13], Chagas et al. apply Genetic Algorithms and Simulated annealing using RSSI values from sensors to identify their location. Other schemes make use of graph-drawing algorithms and produce good results [14], typically using

Kamada-Kawai or Fruchterman-Reingold. Kamada-Kawai utilises spring force [15], whereas Fruchterman-Reingold uses an opposed force-directed algorithm [16]. In [17], Nawaz et al. detail an anchor-free localisation mechanism that utilises a modified graph-drawing algorithm. The approach is based on the Kamada-Kawai graph drawing algorithm [15], utilising a sensor equipped with range-finding devices.

Genetic Algorithms (GAs) have also been used for localisation. GAs require the specification of a fitness function; a mechanism to evaluate a given outcome's suitability. They are a type of evolutionary search heuristic that model natural selection based on fitness. In [18], Zhang et al. detail the implementation of a GA to identify a node's position in a bounded two-dimensional space. The mutation of individual node position is bound by their current location, and the fitness function rewards the correct placement of nodes with respect to their neighbours.

C. Summary

GHT networks, to fulfil their primary purpose, require an identity that provides a node with the opportunity to conduct greedy forwarding. If we are also to use the GHT for distributed data storage, we need the address space to be evenly distributed across the nodes in the topology.

If the protocol was to use GPS sensors, the network would be bound to those identities, meaning that, in order to obtain even data distribution, it would be required that the physical nodes be positioned in a grid layout; unrealistic for many scenarios. Equally, the localisation scheme could provide a relative or anchored location; though this does nothing to address the underlying location dependency.

A scheme is required to allocate addresses that are relatively positioned to maintain reachability but which are distant enough to provide equal coverage of the two-dimensional bounds of the DHT. The next section will detail an approach to a) create node identities that retain their position in relation to their neighbours and also b) provide even coverage of the identity space. This is referred to as a position-relative topology.

IV. Designing the EBL-GHT

The purpose of this work is to design and evaluate a mechanism that will allocate a unique identifier to the individual nodes within the target sensor network. The ID allocated to the node will enable it to take part in a position-relative GHT (using greedy forwarding). The importance of the relationship between the identity and its physical position relates to the ability of the nodes within the network to use a multiple keying function that separates the placement of data within the DHT.

$$K = f(P)$$

$$P_n = \sum_{n=0}^{p} \sum_{k=0}^{q-1} (p_k)_{\substack{md5(uri)\%xmax + \frac{xmax}{d}q\%xmax, \\ md5(uri)\%ymax + \frac{ymax}{d}q\%ymax}} \tag{1}$$

Equation (1) specifies a key rotation algorithm, where P nodes can create keys k based on the MD5 hash value of each node's Universal Resource Identifier (md5(uri)), bound by the dimensions of the key space (xmax and ymax) and distribution count (d). If the data is distributed at a number of points across two dimensions within the DHT - and the DHT's space relates to the physical space - this would lead to better redundancy given localised failure and redundancy no worse in the case of random failure, as found in previous work [19]. It is the intention that the scheme be used both in industrial and home settings, where it would be useful that the state of devices in the network is preserved in the event that a proportion of the system is lost.

For example, in a home IoT network, an oven could be left on, causing a house fire. The information relating to how the fire started could be passed into the remaining network and, depending on the extent of the damage to the building, the state and spread of the fire could be modelled to provide information relating to the cause, thus facilitating a reduction in the risk of future incidents. Alternatively, the information could be extracted in real-time to provide essential information to first responders. This would require that the responders know the mechanism used to key information and the URI used by the individual devices.

As described in the previous section, providing the exact location of a node via GPS is costly and can be difficult to implement indoors. In an effort to overcome the lack of an exact geospatial address, we can approximate the relative positioning between nodes given a complete map of the wireless network. Other research shows that the Fruchterman-Reingold algorithm [16] is a good approach to estimating the individual node positions within a network given a set of edges, vertices and weights. The algorithm provides a similar layout to the target physical topology; an assertion extensively tested by Efrat et al [14].

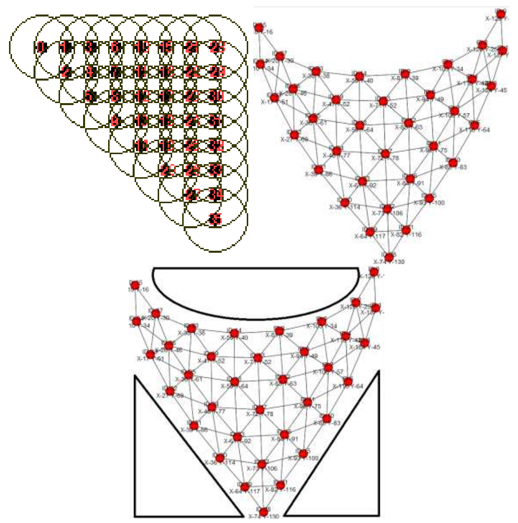

Fig.2. Fruchterman-Reingold localization – This collection of diagrams shows the relationship between a node's physical position (top left) and the Fruchterman-Reingold estimation. Shapes in the bottom diagram show "gaps" in this notional address space.

Fig. 2 shows the original node position (top left), the Fruchterman-Reingold-based layout using the node and neighbour relations with equal weighting on node links (top right). The bottom diagram gives an indication of the poor distribution of identities; creating three voids in the address space, represented by the three shapes. Data falling in these "spaces" would overload the adjacent nodes.

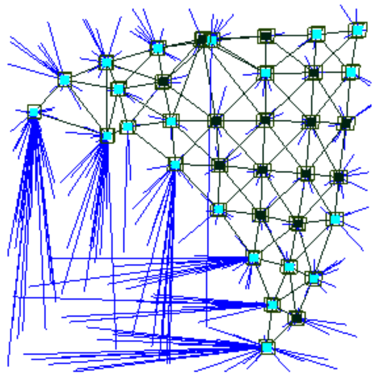

Fig. 3. Address space and key distance. Blue lines approximate the direction and distance between a key's notional location in address space, and the actual node where it is stored. Many long lines, as here, indicate non-optimal placement and node overloading.

Fig. 3 shows the effects of this problem (using the igraph library [20]). We see the placement of keys in a virtual address space; following the identity assignment provided by the Fruchterman-Reingold algorithm. The notional/virtual address space is represented by the whole square while the nodes assigned address space occupy the top right triangle. The blue lines indicate both the direction and distance between where a key is located in notional address space, and the physical node where it has actually been persisted.

As can be seen, half of the nodes in the topology store over half of the keys; with those nodes along the edge between the real and virtual address space particularly overloaded. If a proportion of those nodes are lost, a disproportionate amount of the distributed data would also be lost. An ideal visual representation would be one with few, short blue lines.

Considering the coordinates for the nodes' initial position, the virtual topology can be tested using simulation to determine the total key space imbalance. This metric provides a measure of fitness that can be used to draw comparisons against attempts to create an improved topology. However, the balance is not the only important measure; the topology must also be evaluated to ensure that data items are reachable by corresponding nodes using greedy forwarding. Importantly, if greedy forwarding fails, routes can still be evaluated using the techniques discussed in GPSR [21]. Device longevity can be increased by reducing the requirement for nodes in the network to perform the calculation required to identify a route when greedy forwarding fails.

Therefore, it can be stated that a good candidate topology is one that that nodes are provided with an address relative to its position in the network. This will maintain the ability to use simple greedy forwarding of data and provide nodes with an even share of the total distributed storage requirement of the network.

To solve this multi-criteria problem, the use of Genetic Algorithms will be evaluated to provide a better solution than is provided by the Fruchterman-Reingold algorithm in isolation.

A. The role of Genetic Algorithms

The search for an optimal state through self-organisation is distinct from any higher system intent or goal. Genetic Algorithms are reliant on a) a properly specified fitness function, and b) the environment to dictate the fitness of individuals to thrive and prosper.

Global optimisation looks to find the best possible elements within the set of all possibilities evaluated by a set of criteria; this is referred to as the set of objective functions. The objective function evaluates the current genome population of the model. Genomes can then be ordered by their fitness, identifying those candidates that are closer to the optimum. Once ordered by fitness, individual genomes can be selected to reproduce through the application of crossover and mutation. This process continues bound by time, evolution limit, improvement heuristic measure or through the genomes reaching a certain optimisation threshold.

The solution is therefore deemed the best possible subject to the bounding criterion, which does not necessarily result in the best solution. Usually, evolutionary algorithms are used when the search space is large, and it would therefore be unfeasible to use an analytical solution or there exists no analytical solution to the problem space. This work therefore examines the use of genetic algorithms, themselves a type of evolutionary algorithm; with the objective to solve the address space balancing problem seen in position-relative identity spaces where identity is mapped to a coordinate space.

Genetic Algorithms can utilise multiple objective functions that might specify naturally opposing criteria. In the case of this work – and Wireless Localisation for distributed storage - these functions concern a) the distribution of data relative to their intended location and, b) the ability to successfully use greedy forwarding to locate and retrieve data.

B. EBL-GHT

This section details our novel address localisation scheme. It utilises Genetic Algorithms to generate optimised position-relative topologies for use in GHT.

EBL-GHT is a centralised algorithm; individual nodes distribute neighbour information to a central node that has access to the computational resource that is sufficient to execute the GA. Once the topology has been generated, nodes are provided with a position-relative identity by this central node. The reasoning behind a centralised approach is twofold; firstly due to the need for global knowledge, and secondly the computational overhead of the algorithms involved.

1) Initialisation

EBL–GHT is intended to run alongside existing IoT protocols, e.g. 6LowPAN; providing a redundancy mechanism for M2M communications, or as an alternative to the client-server model found in 6LowPAN. When running alongside Internet Protocol schemes, nodes will utilise the IP address/service identifier tuple to access services distributed within the GHT. Initially, nodes will be identified by a random number that is used for topology construction. Following the position-relative (PR-DHT) construction phase, nodes will be assigned a position-relative two-dimensional GHT address.

In order to start the initialisation procedure, nodes on the network will receive a broadcast from a central coordinating node. For redundancy, there could be multiple coordinating nodes, with individual nodes only responding to a single coordinator. This can be achieved by nodes selecting the coordinating node with the largest ID. The broadcast will be re-sent by each node in the network where a depth counter will be incremented and forwarded by each node in the network; this will create a distributed tree rooted at the coordinator node. Individual nodes will have generated a random identity and transmit initial "HELLO" messages to their connected neighbours, which will facilitate each node building an immediate neighbour table. Nodes will transmit their neighbour table back to the coordinating node. Upon the coordinating node receiving the neighbour maps from all nodes in the network, it can then reconstruct the entire topology. This completes Stage 1 of the protocol.

In Stage 2, the graph is processed by the Fruchterman-Reingold algorithm to establish the physical node positions. The implementation utilises the Fruchterman-Reingold algorithm from the igraph library [20] as it provides a good representation of the original network.

In Stage 3, the layout is rotated through 360 degrees to find the minimum bounding box. Subsequently, a border is added based on the average distance between neighbours and transposes the layout to the required key size. At this point, the virtual unbalanced topology has been constructed, with each node assigned an address within the key boundary. In an effort to balance the topology, it will be passed into the genetic algorithm where it will be evolved to provide a better balance. The resulting topology will provide no worse reachability than is provided by the topology at Stage 3.

2) GA-directed evolution of a force-directed topology

This section will detail the application of Genetic Algorithms to balance the localised topology created in Stage 3. This research has shown that genetic algorithms can be used to find a better solution than the worst case in large search spaces within a bounded search time. Another important aspect of Genetic Algorithms is their ability to fuse multiple metrics to find an optimum that satisfies multiple vectors. The important metrics in this case are the reachability count and the total key space distance imbalance.

The reachability metric I, as shown in (2), is defined as the difference between the expected total number of keys K_i to be stored by each node and the actual return count R_i from each node in the topology querying each saved state of every other node n in the topology.

$$I = \sum_{i=0}^{n} K_i - \sum_{i=0}^{n} R_i \qquad (2)$$

The key space imbalance metric, as detailed in (3), is the total displacement D of the virtual two-dimensional coordinates of all data items d(x,y), currently stored in the DHT at nodes n(X,Y) from the virtual node coordinate that each of the data items are being stored.

$$D = \sum_{i=0}^{n} \sum_{j=0}^{d} \sqrt{(X(n)_i - x(d)_j)^2 + (Y(n)_i - y(d)_j)^2} \qquad (3)$$

It is normal practice that a Genetic Algorithm is seeded using a random population; however, through experimentation, it has been determined that a random population fails to evolve and form a network that meets the fitness and reachability of the initial layout algorithm provided at the end of Stage 3. The approach taken in this work seeds the Genetic Algorithm with the coordinates provide by Stage 3, which provides the population with a better than random starting point. In Stage 4, the x and y coordinates of the Stage 3 topology are encoded into the population P chromosome of ten candidates. Two chromosomes ($C_1 C_2$) are treated as parents and 8 ($C_3 .. C_{10}$) as children, applying mutations to each of the 8 in pairs ($p_2 .. p_5$) using the following mutation: levels $M = Log(nodecount)$ ^2 .

The mutations are limited to a random co-ordinate within four radius levels surrounding the existing coordinate in the chromosome; this allows the search space to be gradually expanded. The candidate co-ordinates are then copied into the simulator model. The model initialises, enabling the nodes to form a PR-DHT. Each node then saves its own state using a three-position replication strategy, utilising the rotation algorithm shown previously in (1). Once all save operations have been completed, each node is requested to retrieve every saved state within the system. Once this is complete, each genome is scored using the key displacement metric and sorted with the lowest key displacement score being awarded position 1 and the highest being placed in position 10. Any chromosome that has a saved key return metric any worse than the best will not be processed; this helps to ensure that the reachability of the topology improves or remains constant for each evolution.

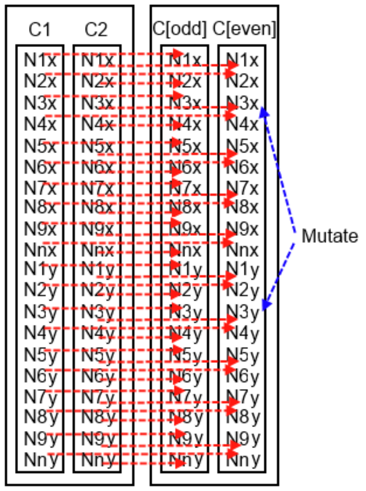

Fig. 4. Crossover and mutation – This shows the mutation strategy for the evolution of the topology. Note the crossover of the node coordinates and application of mutation to coordinate sets.

Children ($C_3 .. C_{10}$) are subsequently populated using genomes ($C_1 C_2$), applying an interleaving crossover, as shown in Fig. 4, that alternates with an even-and-odd-node position as the starting chromosome entry. Mutation is applied, and the process evolves for a set number of generations. The mutation function generates a random coordinate within a fixed radius of the original coordinate, which reduces the opportunity for the network to get stuck in a local optimum. This could otherwise occur by finding a poor candidate that satisfies an initial low reachability metric and improves on the key space distance metric. Upon completion of the mutation phase, the process of testing the topology starts again, which is repeated until the expiration of the time constraints placed upon the process, or until the topology reaches a steady state; that is, no improvement on the score for a set number of evolutions.

When the genetic balancing phase is complete, the new identities are distributed to the devices in the network. In order to reduce the overhead associated with broadcasting each ID, a minimum spanning tree is drawn over the topology and traversed, transmitting the ID of the node and the required neighbour IDs to complete the tree. Neighbours will populate the remaining neighbours using local neighbour broadcasts; enabling the protocol to limit the packet size required.

3) Alternating fitness functions to improve performance

As described in the previous section, two metrics have been defined to evaluate the suitability of the evolved network. They each provide a fitness evaluation in terms of a) reachability and b) key displacement/imbalance, respectively.
It is difficult to combine parameters that are not coincident. For example a company's profitability is not usually coincident with its employee wellbeing metric. To provide a suitable topology we evolve, alternating between different fitness functions, testing for reachability for ff generations and then key displacement for ff generations. This provides an equal timeslot for each function to influence the topology as well as limiting the effect that any individual function has within a single evolutionary phase.

V. Evaluation

This section will evaluate the application of Genetic Algorithms to the localisation of nodes within a wireless sensor network. This section begins with a brief overview of the simulation environment used for all evaluative data-gathering runs. The following two sub-sections then provide empirical justification for two defining features of this work; the non-random approach to GA seeding, and the application of an alternating-target fitness function influencing the GA's evolutionary path.

This section concludes with a review of the capability of the EB-GHT to create balanced position-relative identity spaces across a range of test topologies, followed by a brief evaluation of the execution time implications of this approach.

A. Simulation Environment Overview

The results presented in this section have been generated by a Python Discrete Event Distributed system simulator developed for this work. Simulation runs were executed on Ubuntu 12.04 running on an Intel i5-4200U (Haswell-ULT) 1.6 GHz machine, with 4GB of DDR3 RAM.

The simulator permits us to simulate the networked environment where nodes are placed physically, corresponding to five different layouts in two different sizes. Nodes operate independently with their own isolated state machines; there is no global knowledge provided to devices. Communication between nodes is achieved through input packet buffers. Where the simulator identifies nodes in the communication range, a buffer relationship is established, thus enabling them to communicate.

The simulator is able to drive a genetic algorithm through a series of evolutions where a population of 10 test networks are created, made up as follows:

To test the robustness of the approach, 5 different topology types have been created with 2 size variations for each topology. These topologies are shown across Fig.5 (Triangle, H, Square, and Hole Shapes) and Fig. 6 (L-shaped). The topology types have been chosen due to their range of local-minimum-inducing features. The square grid pattern is intended to give ideal results, both in terms of the geographic node dispersion, and the opportunity for the Fruchterman-Reingold algorithm to produce a topology with excellent characteristics.

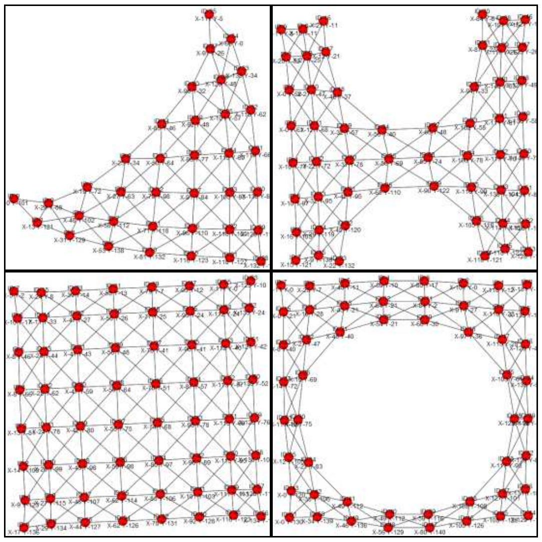
Fig. 5. Simulation topologies used (clockwise from top left) Triangle, H, Hole, and Square. The square topology acts as a reference, whereas the other topologies have local-optimum-inducing features

This serves as both a benchmark and a mechanism to validate the simulator. The square topology provides a comparative measure for the other topology types; a source of exemplar results for reachability, balance and key distance. Each topology is evolved through a total of 200 cycles. These networks simulate the storage of 9 keys per node, then retrieve the entire key space. In each evolution step, the coordinates of the nodes are mutated according to the method described in the previous section.

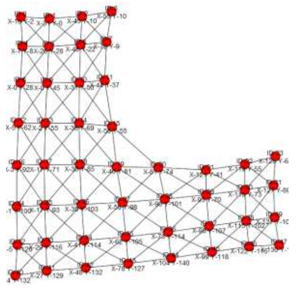

Fig. 6. L-shaped topology used in GA seeding and fitness function simulations.

B. Using Fruchterman-Reingold to Seed the GA

It is common to populate an open-search-space genetic algorithm with a random initial distribution and then evolve to an optimum; however, in this work, the genetic algorithm is seeded with a Fruchterman-Reingold graph layout of the topology. The use of graph-drawing to address the problem of localisation for nodes without GPS devices has been shown to be effective in other work on sensor localisation [22].

However, in order to validate this decision for this work, comparative experiments were undertaken with random seeding against layout algorithm seeded GAs. Firstly, the GA is seeded with a randomly-arranged 16 nodes. The network is then evolved for 100 cycles and its fitness measured. This experiment is then repeated under the same conditions but seeded using the Fruchterman-Reingold (FR) layout. The results in Table 1 identify the advantages of using the FR approach.

TABLE 1
FRUCHTERMAN-REINGOLD VS RANDOM SEEDING

	Fruchterman-Reingold	Random
Start State Loss	0	74
End State Loss	0	44
Start KL	244	1457
End KL	0	1355
Start KD	4282.97	4726.81
End KD	2291.41	4138.56
Start MSD	4.63	5.13
End MSD	3.5	4.75

KL = Individual Key Loss, KD = Key Displacement, MSD = Maximum Store Deviation; (Values to 2dp; lower values are better)

The initial FR approach has a total state loss of 0; after 100 cycles, the Random topology still has a total loss of 44, having started with a total loss of 74. The individual key loss shows 0 at the end of 100 cycles for the FR layout, with 1,355 individual keys for the Random topology. Key displacement and key deviation both show similar properties, being worse in the random initialisations over the FR approach. Such results provide justification for the decision to start with a topology that is pre-localised rather than a random one.

C. Evaluation of the Alternating Fitness Function

To recap; the joint aims of EBL–GHT are a) to create an identity scheme that improves the reachability of nodes using greedy forwarding as this will require fewer devices to employ a routing algorithm to circumvent local minimum, whilst b) at the same time improving the topology balance so that items that are saved into the topology are evenly distributed.

The approach taken in this work is thus centred on using two fitness functions; one to evaluate the total distance that keys are saved from their ideal location and one to measure reachability. The latter is calculated as the total number of nodes that are able to see every state from every device on the network.

In the simulated environment, the performance of the GA is measured with the 2 fitness functions applied separately, jointly, and finally applied alternatively – switched each ff generations. For the purposes of the latter experiment, ff was 5; such that the fitness function alternated every 5 generations. The simulations in this section were carried out using the L-shaped topology pattern, shown in Fig.6. Each test was carried out across 200 generations.

Table 2 shows the resulting performance of these approaches. Alternating fitness on key loss and distance provides the greatest reduction in key loss, and also the greatest reduction in key displacement. Accumulating key loss and distance provides the worst key loss difference but also provides good maximum deviation values for data-store displacement.

TABLE 2
FITNESS FUNCTIONS TEST

	Alternating KL / KD Fitness	Accumulated KL/ KD Fitness	KL Fitness	KD Fitness
Start KL	244	184	189	314
End KL	0	145	97	215
KL Difference	244	39	92	99
Start KD	4282.97	3563.14	4036.78	3968.01
End KD	2291.41	2259.01	3350.73	2341.93
KD Difference	1991.56	1304.13	686.06	1626.08
Start MSD	4.63	4.75	5	5.25
End MSD	3.5	2.86	4.63	4.75
MSD difference	1.13	1.88	0.38	0.5

KL = Key Loss, KD = Key Displacement, MSD= Max Store Deviation (Values to 2dp; lower values are better).

Testing key loss alone provides poor results in all tests, whereas testing key distance fails to complete with the alternating approach. Accumulating key loss and distance provides a better deviation but, owing to the comparatively poor reachability score (i.e. lost keys), the results show that alternating between key loss and distance provides the better approach.

Fig. 7 shows the network's performance during the alternating fitness evolution; the bottom plot represents key loss, while the top plot represents data displacement. The network's current fitness measure is shown by the blue over-plot:

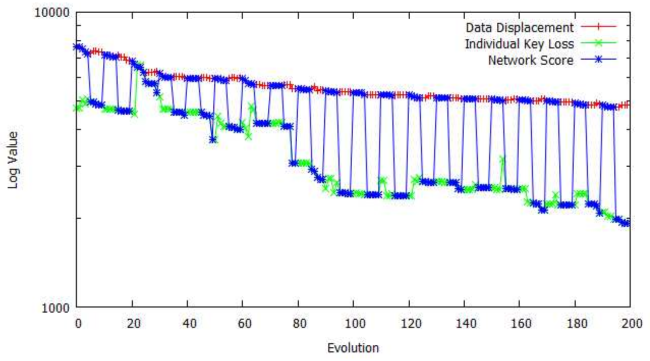

Fig. 7. Alternating fitness function and effect on network score; the current fitness selection, (the blue line), is shown to switch between key displacement and key loss.

D. Alternating fitness function effects

This section discusses the evaluated capability of EBL–GHT in creating position-relative identities for devices in Geographic Hash Tables. Each generated topology is assessed to measure how well it fulfils the following requirements:

Reachability / Key Loss (KL). Reduce the total number of states that devices in the network are unable to retrieve. To provide an assessment of reachability, each node saves a set number of keys into the topology. Following the completion of the save operation, each node on the network retrieves each of the save states from every other device in the topology. Upon receipt, each node keeps track of the number of returned states. Following the completion of the save and retrieval steps, the number of keys that have been distributed and the number that should have been returned are calculated. The network score is based on the number of missing keys.

Key Distance (KD). As described in Section IV and Fig. 3, the notional available key space can be represented by a 2-dimensional space. The actual assigned or persisted location of a key is likely to, owing to the topological properties and available key space, differ from its ideal notional location. As such, the EBL-GHT should reduce the total distance that all keys in the network rest from their intended destination in the key space. Initially, nodes are provided with an identity that is relative to their position in the network. Data will then route to a point that is closer to the target, eventually reaching a node that is closest (best case) or hitting local minimum and saving on a device that is not the closest to the destination coordinate (worse case). If a node has a disproportionate quantity of the address space, more keys will be placed on this node. Some keys will have been intended for the location that the node occupies, whereas other data would have been destined for coordinates that are distant from the device. The probability that a network has zero key space distance is extremely low. It would require an infinite number of devices spread out equally across a square area or the data items keying exactly to the address of nodes in the topology evenly.

Maximum data store deviation. Reduce the storage imbalance in the network. This is achieved by calculating the mean store size across the nodes, and then calculating the maximum deviation. We look to reduce the maximum deviation to distribute data more evenly across the network.

These criteria are first assessed in the smaller set of reference topologies. The results from these experiments are shown in Table 3. The small topology patterns shown earlier in Fig. 5 and Fig. 6 have limited node counts and operate within a smaller network boundary of 100x100 metres. Experiments showed that this limits their opportunity to improve storage balance, and a small change can have a very pronounced effect (positive or negative) on this balance.

TABLE 3
SMALL TOPOLOGY RESULTS

	Square (25)	L Shape (16)	H Shape (19)	Triangle (15)	Hole (21)
Starting KL	0	244	562	132	458
Ending KL	0	31	3	0	182
KL difference	0	213	559	132	276
Starting KD	1759.9	4282.9	2338.1	2323.2	2384.6
Ending KD	1508.7	2291.4	1787.9	1367.9	1906.3
KD difference	251.2	1991.6	550.1	955.3	478.3
Starting MSD	1.8	4.6	3.3	5.9	2.5
Ending MSD	2.5	3.5	2.4	3.2	3.1
MSD difference	-0.64	1.1	0.8	2.7	-0.6

KL = Key Loss, KD = Key Displacement, MSD= Max Store Deviation; (Values to 1dp; lower values are better).

However, the results of our experiments show that all topologies do still improve their reachability, with the 15-node triangle topology achieving full reachability. The L-shape topology with 16 nodes, the H-shape topology network with 19 nodes and the Hole topology with 21 nodes are all missing a number of states. The topologies still suffering missing states are those most likely to exhibit local-minimum-inducing features. Notably, however, they do show excellent improvements over the use of Fruchterman-Reingold alone, with an improvement in the reachability of 559 states for the H topology, 276 for the Hole-shaped topology and 213 for the L-shaped topology. This would reduce the number of route calculations required to identify alternative paths to establishing those data elements not accessible by greedy forwarding alone.

The final key deviation for the square topology with 25 nodes is worse that the initial topology. However, at evolution 118, the results show a maximum deviation of 1.3 with no missing keys. The circular hole-shaped topology with 21 nodes also has a worse maximum deviation. However, as with the square topology, a previous generation had a better deviation with a score of 1.8 with an equivalent missing key count.

Therefore, in order to obtain the best results, it is possible that saving historical best deviation and missing key count and comparing that to the final evolved topology may be the best approach.

The results in Table 4 show the effect of EBL-GHT on larger topologies bound by an area of 160x160 metres. As was seen in the small topology, the square with 64 nodes provides excellent reachability and storage balance. The square topology provides a reachability loss count of 0 before and after the 200 evolution steps. The topology also exhibits a slight improvement in storage balance. The L-shaped topology with 48 nodes shows an improvement but an increase in key imbalance.

TABLE 4
LARGE TOPOLOGY RESULTS

	Square (64)	L Shape (48)	H Shape (54)	Triangle (36)	Hole (52)
Start KL	0	1972	4737	2626	5760
End KL	0	1397	1921	576	3895
KL difference	0	575	2816	2050	1865
Start KD	4118.3	7482.3	7602.3	6683.7	8679.5
End KD	3756.4	5042.2	4850.9	4144.6	6455.1
KD difference	361.9	2440.2	2751.4	2539.1	2224.4
Start MSD	3.0	4.9	5.1	6.4	4.7
End MSD	2.8	5.3	4.5	4.2	3.7
MSD difference	0.3	-0.4	0.6	2.2	0.9

KL = Individual Key Loss, KD = Key Displacement, MSD = Max Store Deviation; (Values to 1dp; lower values are better)

As was observed through the small topologies, a previous evolution had improved balance characteristics, achieving a balance of 4.4. However, this time, the key loss is slightly higher with a loss of 14,000 keys at evolution 107. There is also a deviation of 4.05 with a key loss of 2,000, which leads to decisions as to the importance of reducing key balance against the cost of key reachability. The H-shaped topology with 54 nodes, the Triangle topology with 36 nodes and the Hole topology with 52 nodes all improve their key counts. The H, Triangle and Hole topology also improve their key imbalance.

E. Execution Time and Performance Implications

The previous sections have illustrated the quality of resulting localisation and position-relative identities generated by the proposed approach. This section will discuss some of the performance implications, starting with a review of the execution time for EBL-GHT by network size. The results over a variety of sizes are presented in Table 5:

TABLE 5
GA EXECUTION TIMES

Total Nodes	Single Evolution Time	Total Time (200 cycles)
25	2.9s	580s (9’40”)
36	7.4s	1480s (24’40”)
49	16s	3200s (53’20”)
64	31s	6200s (103’20”)
81	56s	11200s (186’40”)
100	95s	19000s (316’40”)

(As described in Section IV, during a single evolution, 10 candidate networks are evaluated.)

The GA execution time is bound by node count, rather than topology. The results shown in the table show an exponential increase in runtime against size; runtime can be affected by a reduction in candidates in each evolution. For the scenario given – a long term deployed network with infrequent or periodic changes – a lengthy predicted runtime such as this is not problematic. EBL-GHT could provide bootstrap / refresh functionality and used as a response to triggers such as growing key imbalance.

However, in a rapidly changing or unstable system, a non-trivial execution runtime is likely to be unsuitable. Whilst beyond the scope of this paper, optimisations and further localisations could minimise the execution times for changes to an existing network. For example, providing the boundary topology of the network is unchanged, addition or removal of nodes could be treated as an incremental change and identities allocated locally and in-network.

TABLE 6
GA PACKET OVERHEAD

Nodes	Initialisation / neighbour data (packets)	Position relative address allocation (packets)	Total overhead (packets)
25	140	40	180
36	244	100	344
49	392	196	588
64	592	336	928
81	852	528	1380
100	1180	780	1960

(Initialisation and distribution packet cost quantifies the packet overhead in initialising and distributing neighbour data and the root anchored tree)

The results provided in Table 6 show the number of packets transmitted during the initialisation phase. This includes the generation of each node’s neighbour table, the construction of the root anchored tree and the distribution of neighbour tables to the root node.

The packet cost for position-relative address allocation quantifies the cost in the root node allocating every other node its address. As the resulting network requires routing, this includes the overhead of each multi-hop retransmission.

This sort of overhead is typical of a multi-hop network, which relies on (or relays data to) a centralised co-ordinating node. This work has not considered optimising this overhead, but a degree of improvement could be attained by aggregating data when nodes transmit the neighbour map back, and in address allocation. However, in this case, care would have to be taken in managing neighbour map dimensions to avoid overrunning frame size.

The following section identifies some further avenues for investigation regarding performance implications and optimisation in future work.

VI. SUMMARY, DISCUSSION AND FUTURE WORK

This paper detailed a novel technique for generating balanced position-relative identities for use in Geographical Hash Tables in the IoT. The approach, EBL–GHT, utilises a novel alternating fitness function combining the benefits of two metrics to improve an initial topology. This work also introduces the novel technique of evolving based on the output of wireless network simulation to generate layer topology improvements.

The simulation results have shown that the approach detailed improves on the initial Fruchterman-Reingold layout for all layout types; both for key loss and store-size deviation. This will reduce the total power consumed by the network when making greedy forwarding decisions, reducing the requirement of the angle calculations needed by local minimum avoidance techniques. The reduction of imbalance in data-store size results in a fairer distribution of state amongst individual devices.

In summary, the following conclusions have been established through this study and experimentation:

Force-directed layouts and topology balance. The Fruchterman-Reingold graph layout used in isolation produces a good representation of a target topology given the neighbour relationships of all devices. However, the topology alone provided poor balance and reachability when using rendezvous on top of a Geographical Routing Protocol using Greedy Forwarding.

Evolution of force-directed layouts. Using a Genetic Algorithm, it is possible to evolve the initial coordinates provided by the Fruchterman-Reingold algorithm. This improves balance and reachability using rendezvous communications on top of Geographical Routing.

Alternating fitness functions. Adopting an alternating fitness function approach can improve the ability of the GA to find an optimal solution, when compared to those metrics being used alone or through aggregation of metrics.

This research study opens up a rich seam of potential future work. There is scope for immediate future work in terms of further improving the efficiency and efficacy of the GA approach. Clear performance gains have been achieved by the use of an alternating fitness function, over a crude aggregation of both fitness metrics. However, there is work to do in terms of analysing the ff variable tuning, and in turn, the fitness “switchover”.

Furthermore, in terms of performance, while the GA can easily be bounded by the number of generations it will evaluate; it is still a time consuming process. Building each test network and evaluating it is computationally costly. As such, while the process has use in terms of an initialisation or periodic network (re)structuring operation, it is not realistic to consider it a mechanism for real-time adaptation. Future work will investigate these issues of timeliness; including further efficiency improvements and parallelisation approaches. Specifically we will look to add stochastic network planning capabilities to low power devices using hybrid integrated SOC + FPGA technology. The authors believe that the many billions of devices that will be deployed to the Internet of Things will require in-network/on-silicon solutions to stochastic network planning and operations.

VII. REFERENCES

[1] P. Maué and J. Ortmann, “Getting across information communities,” Earth Sci. Informatics, vol. 2, no. 4, pp. 217–233, Nov. 2009.

[2] Z. Shelby, K. Hartke, and C. Bormann, “Constrained Application Protocol (CoAP),” IETF, 2012.

[3] T. Winter and P. Thubert, “RPL: IPv6 Routing Protocol for Low power and Lossy Networks.,” Internet Draft Draft. Work Prog., 2010.

[4] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, and S. Shenker, “GHT: a geographic hash table for data-centric storage,” in Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications - *WSNA ’02*, 2002, p. 78.

[5] A. Awad, C. Sommer, R. German, and F. Dressler, “Virtual Cord Protocol (VCP): A flexible DHT-like routing service for sensor networks,” in 2008 5th IEEE International Conference on Mobile Ad Hoc and Sensor Systems, 2008, pp. 133–142.

[6] A. A.-B. Al-Mamou and H. Labiod, “ScatterPastry: An Overlay Routing Using a DHT over Wireless Sensor Networks,” in The 2007 International Conference on Intelligent Pervasive Computing (IPC 2007), 2007, pp. 274–279.

[7] H. Pucha, S. M. Das, and Y. C. Hu, “Ekta: an efficient DHT substrate for distributed applications in mobile ad hoc networks,” in Mobile Computing Systems and Applications, 2004. WMCSA 2004. Sixth IEEE Workshop on, 2004, pp. 163–173.

[8] D. A. Bader, V. Agarwal, and K. Madduri, “On the Design and Analysis of Irregular Algorithms on the Cell Processor : A Case Study of List Ranking - Georgia Institute of Technology,” Cell, 2007.

[9] S. Ratnasamy, B. Karp, S. Shenker, D. Estrin, R. Govindan, L. Yin, and F. Yu, “Data-Centric Storage in Sensornets with GHT, a Geographic Hash Table,” Mob. Networks Appl., vol. 8, no. 4, pp. 427–442, Aug. 2003.

[10] M. Albano, S. Chessa, F. Nidito, and S. Pelagatti, “Dealing with Nonuniformity in Data Centric Storage for Wireless Sensor Networks,” IEEE Trans. Parallel Distrib. Syst., vol. 22, no. 8, pp. 1398–1406, Aug. 2011.

[11] C. E. Perkins and P. Bhagwat, “Highly dynamic Destination-Sequenced Distance-Vector routing (DSDV) for mobile computers,” ACM SIGCOMM Comput. Commun. Rev., vol. 24, no. 4, pp. 234–244, Oct. 1994.

[12] D. Niculescu and B. Nath, “Ad hoc positioning system (APS),” in *GLOBECOM’01. IEEE Global Telecommunications Conference (Cat.* No.01CH37270), 2001, vol. 5, pp. 2926–2931.

[13] S. H. Chagas, J. B. Martins, and L. L. de Oliveira, “Genetic Algorithms and Simulated Annealing optimization methods in wireless sensor networks localization using artificial neural networks,” in 2012 IEEE 55th International Midwest Symposium on Circuits and Systems (MWSCAS), 2012, pp. 928–931.

[14] A. Efrat, D. Forrester, A. Iyer, S. G. Kobourov, C. Erten, and O. Kilic, “Force-directed approaches to sensor localization,” ACM Trans. Sen. Netw., vol. 7, no. 3, pp. 27:1–27:25, Oct. 2010.

[15] T. Kamada and S. Kawai, “An algorithm for drawing general undirected graphs,” Inf. Process. Lett., vol. 31, no. 1, pp. 7–15, Apr. 1989.

[16] T. M. J. Fruchterman and E. M. Reingold, “Graph drawing by force-directed placement,” Softw. Pract. Exp., vol. 21, no. 11, pp. 1129–1164, Nov. 1991.

[17] S. Nawaz and S. Jha, “A Graph Drawing Approach to Sensor Network Localization,” IEEE Int. Conf. Mob. Adhoc Sens. Syst. Conf., vol. 0, pp. 1–12, 2007.

[18] Q. Zhang, J. Wang, C. Jin, J. Ye, C. Ma, and W. Zhang, “Genetic Algorithm Based Wireless Sensor Network Localization,” in 2008 Fourth International Conference on Natural Computation, 2008, vol. 1, pp. 608–613.

[19] A. Attwood, O. Abuelma’atti, and P. Fergus, “M2M Rendezvous Redundancy for the Internet of Things,” in Sixth International Conference on Developments in e-Systems Engineering - DeSE2013 (DeSE2013), 2013.

[20] G. Csardi and T. Nepusz, “The igraph Software Package for Complex Network Research,” InterJournal Complex Syst., p. 1695, 2006.

[21] B. Karp and H. T. Kung, “GPSR,” in Proceedings of the 6th annual international conference on Mobile computing and networking - *MobiCom ’00*, 2000, pp. 243–254.

[22] S. Nawaz and S. Jha, “A Graph Drawing Approach to Sensor Network Localization,” in 2007 IEEE Internatonal Conference on Mobile Adhoc and Sensor Systems, 2007, pp. 1–12.

Andrew Attwood gained his PhD in 2014 from LJMU, and an MSc in Advanced Computer Science from the University of Manchester in 2010. He has worked in education and training, providing services to International Telecommunications companies. Andrew has also been successful in a recent UK TSB (SBRI) grant funding application for Future Cities data platforms.

He is presently employed as a Lecturer in the School of Computing, Mathematics and Digital Technology, Manchester Metropolitan University. His research interests include Distributed and Parallel Computing, Evolutionary Algorithms and Wireless IoT technology.

David Lamb gained his PhD in 2009, and a BSc (Hons) in Software Engineering in 2005, both from LJMU. Before academia, David worked as a Software Engineer for several large UK-based software houses, and progressed on to providing development and consultancy services directly to organization in the public and private sectors. He is presently a Senior Lecturer in the School of Computing and Mathematical Sciences at LJMU. David presently serves on the TPC for the IEEE Conference Developments in eSystems engineering, and his research interests include work in Distributed Software Engineering and support for large-scale software.

Omar Abuelmaatti received a PhD in Network Communication from LJMU in 2006 and an MSc in Data Telecommunications and Networking from Salford University in 2000. He has been with LJMU since 2001 where he is now a Programme Leader and a member of the Department of Networked Systems and Security and the Distributed Multimedia Systems and Security Research Group. He has participated in a number of successful research and development projects. He has extensive expertise in wireless and mobile communication systems, including Bluetooth, ZigBee and WLAN, GSM, UMTS and WiMAX. His current research interests include; Critical Infrastructure and Smart Networks; Wireless and Mobile Computing; Wireless Sensor Networks and the Internet of Things; and Distributed and
Cloud Computing.