

Comparing Design Of Experiments and Evolutionary Approaches To Multi-Objective Optimisation Of SensorNet Protocols

Jonathan Tate, *Member, IEEE*, Benjamin Woolford-Lim, Iain Bate, *Member, IEEE* and Xin Yao, *Fellow, IEEE*

Abstract—The lifespan, and hence utility, of sensorNets is limited by the energy resources of individual motes. Network designers seek to maximise energy efficiency while maintaining an acceptable network Quality of Service. However, the interactions between multiple tunable protocol parameters and multiple sensorNet performance metrics are generally complex and unknown. In this paper we address this multi-dimensional optimisation problem by two distinct approaches. Firstly, we apply a Design Of Experiments approach to obtain a generalised linear interaction model, and from this derive an estimated near-optimal solution. Secondly, we apply the Two-Archive evolutionary algorithm to improve solution quality for a specific problem instance. We demonstrate that, whereas the first approach yields a more generally applicable solution, the second approach yields a broader range of viable solutions at potentially lower experimental cost.

I. INTRODUCTION

SensorNets compose many autonomous *motes* into ad-hoc networks for distributed sensing and processing applications. Motes are small, cheap computers equipped with independent power supplies, wireless communication capability, and sensors with which to passively monitor with their environment. An important feature of sensorNets is that their resource availability is minimal especially in terms of energy. When sufficient motes have run out of energy for the sensorNet application to stop functioning effectively, the sensorNet is effectively dead. This can occur even if the majority of motes continue to enjoy substantial unused energy reserves.

Wireless communication components are generally the most energy-hungry subsystem of sensorNet motes [1]. Energy efficiency improvements are possible both in sensorNet hardware and software. In this paper we consider only software-driven improvements at the level of network middleware, and assume a fixed hardware platform and typical distributed sensing application. Our results are therefore portable across hardware platforms and high-level software applications in typical sensorNet scenarios.

We demonstrate that careful tuning of networking protocols can yield substantial improvements in energy efficiency without compromising performance. We compare two fundamentally different approaches and show that while each has merit, a broad-but-shallow Design Of Experiments (DOE) approach is more effective at finding high-quality solutions for a given network instance than an Evolutionary

Algorithm (EA) narrow-but-deep approach. To the best of our knowledge, our work is the first to apply EA techniques to the sensorNet protocol tuning problem, and the first to compare its efficacy to that of the DOE approach for sensorNets.

Undertaking this work presented interesting challenges. SensorNet protocol tuning is not a simple, idealised problem. It is a real-world problem with multiple inputs, multiple outputs, and multiple objectives. The complex interrelationships between these factors were not known at the outset, and as such could not be targeted specifically during experiment design. A further complication was the unusually high cost of fitness function evaluation within the evolutionary algorithm. Each evaluation of a candidate solution required a full simulation instance to be executed. Obtaining results of suitable quality in acceptable time required novel optimisations in simulation. Source data obtained by experiment necessarily contains some level of experimental noise which we had to take into account during experiment design. The research objectives addressed in this paper are as follows.

- Obj 1: Obtain near-optimal solutions to the sensorNet protocol tuning problem using Design Of Experiments and evolutionary approaches.
- Obj 2: Compare solution quality attainable by design of experiments and evolutionary search methods for a given network design.

II. RELATED WORK

Tuning network protocols for energy efficiency is a complex problem owing to interactions between parameters. The resulting combinatorial explosion of possible solutions renders exhaustive search impossible [2]. A methodology is proposed based on full factorial experiment design to explore solution space in acceptable time, fitting statistical models to network performance metrics to summarise the relationship between tunable parameters and network response.

The input variable ranges and multiple performance metrics designed by Tate *et al.* [2] define a *Multi-objective Optimisation Problem* (MOP). It is appropriate to apply multi-objective problem solving techniques to identify input parameter sets giving near-optimal sensorNet performance.

A major issue with multi-objective problems is the difficulty in determining a well-defined ordering of solution quality for a given set of candidate solutions. In single-objective problems, better solutions are simply those with a lower fitness. With MOPs, there are multiple fitness values to consider. A solution with every fitness value lower than another is clearly superior, and is said to *dominate* the other solution. However, solutions may be lower in one fitness

B. Woolford-Lim and X. Yao are with the School of Computer Science, University of Birmingham, Birmingham, B15 2TT, United Kingdom (email: b.j.woolford-lim@cs.bham.ac.uk; x.yao@cs.bham.ac.uk)

J. Tate and I. Bate are with the Department of Computer Science, University of York, York, YO10 5DD, United Kingdom (email: jt@cs.york.ac.uk; iain.bate@cs.york.ac.uk)

value but higher in another; this kind of solution is termed *non-dominated* and is harder to classify. In place of an optimum value, MOPs tend to have what is termed a *Pareto-optimal front*, along which all solutions are mutually *non-dominated*, and all other valid solutions are dominated by the PO front members.

Combinatorial explosion of possible solutions within multi-variable problem renders exhaustive search impossible. Stochastic search algorithms explore the solution space non-exhaustively in reasonable time. Multi-objective evolutionary algorithms (MOEAs) are a type of a stochastic generational multi-objective search algorithm. Using techniques inspired by the concept of *survival of the fittest*, they seek to find this Pareto-optimal front by evolving progressively better solutions based on the relative *fitness* of previous solutions. A number of such algorithms have been proposed, including NSGA-II [3], SPEA2 [4], and Two-Archive [5].

III. EXPERIMENT DESIGN

In this section we define the experimental configuration used in sections IV and V to tune a sensornet protocol and explore the parameter landscape.

A. Experimental approaches

Two experimental approaches are employed in this paper. The Design Of Experiments (DOE) approach, using Factorial Design, is used to sample the entire problem space in a systematic and even manner [6]. This is a broad-but-shallow deterministic search method. Then, we implement an evolutionary approach, using the Two-Archive algorithm to sample the problem space in a guided and uneven manner [5]. This is a narrow-but-deep stochastic search method. The DOE part was implemented first and the preliminary results used to define a problem space region in which we predict optimal solutions to reside. We then used this insight to focus the evolutionary portion of the work by providing Two-Archive with initial values in that region. We therefore contend that, for problems of this type, it is valuable to assign some experimental effort to an initial DOE stage to focus a later evolutionary stage. This hybrid approach exploits the strengths of both constituent components.

B. Experimental technique

It is impractical to perform the experiments described in sections IV and V using real networks due to high overheads of logistics, cost and time. Additionally, it is virtually impossible to guarantee a consistent and unchanging environment for the total runtime of the tens of thousands of experiments. This would severely undermine the validity of comparison between results obtained from multiple experiments, which is critical to the analytical methods we propose. To address these concerns all experiments were conducted by simulation, using the YASS multithreaded simulator [7] optimised for sensornet experiments and for this duty pattern.

To ensure accuracy of the derived interaction model, simulations in the DOE portion of the experiment ran for 120 simulated seconds before sampling to allow metrics to

converge. This is considerably in excess of the 10 simulated seconds required for the metrics to converge within $\pm 5\%$ experimental error [2]. For the Two-Archive portion of the experiment the relative fitness of candidate solutions is more important than absolute fitness, and the algorithm is expected to gravitate toward non-degenerate solutions for which metrics converge relatively quickly. Simulations for this method ran for 20 simulated seconds before sampling. This reduced overhead and allowed more generations to be evaluated per unit time. Due to the stochastic nature of network communication, metrics are not guaranteed to be identical between repeats of the same experimental setup. Therefore, all simulations in both the DOE and Two-Archive tests were repeated three times and the results aggregated.

C. Network configuration

The techniques outlined in this paper are independent of the specific protocols and network designs explored in the following experiments. However, these experiments explore only a finite portion of the unbounded design space of all networks and all protocols. It is likely that the trends we identify in network performance responses as a function of protocol tuning parameters are likely to apply in similar networking contexts. Nevertheless, we limit the scope of our claims to the portion of design space defined in this section, within which we have confidence in our findings as they are demonstrated to have statistical significance.

A set of three typical sensornets was defined and reused for all experiments. Each sensornet consisted of 250 static motes of identical capability modelled on the Crossbow MICA2 mote. Motes were distributed randomly within a square of side length 21Km yielding a geographic distribution of uniform planar density. The resulting average *degree of connectivity*, the number of peer nodes with which a given node can feasibly communicate, is approximately 20 which is typical of sensornets [8]. All internodal communication was defined to occur through anisotropic radio broadcast in an obstacle-free vacuum. Signal propagation and attenuation was modelled using the Friis free space model with exponent of 2.0. The simulated motes ran a simulated distributed sensing application in which every node periodically produces a small data packet. The destination of each packet is randomly selected from all motes in the network to prevent bias from any implicit structure in the mote distribution.

An adapted form [2] of the TTL-bounded gossiping protocol [9] was selected to manage in-network communication. This protocol is ignorant of energy, network topology, and the host application, ensuring no bias in the results produced. Flooding and gossiping protocols of this form are commonly used within more complex protocols [9] to establish delivery routes or maintain awareness of network status, widening the scope of our results to all such protocols.

Each node in the network can act as a packet source, a packet destination, or a packet relay. When a source node creates a packet it is queued for broadcast to the wireless medium. If the packet is eventually broadcast it may be received by one or more other nodes within communication

range able to successfully extract the packet data from background noise. Each packet recipient independently determines whether the packet is dropped (based on a *Time To Live* (TTL)), consumed (based on recipient identity), or queued for rebroadcast (based on whether a random value, between 0 and 1, exceeds a threshold - the *gossip probability*). Packet headers specify one or more destinations, defining the only nodes at which a given packet can be consumed. In our experiments we specify exactly one destination per packet. Packet headers also specify TTL in terms of node-to-node hops and lifespan to prevent stale packets circulating indefinitely. Assuming these header-defined limits are not exceeded each recipient makes an independent probabilistic decision whether to rebroadcast the packet to its neighbours. The packet thus radiates outward from the source node, hopefully arriving at least once at each intended destination.

D. Controlled factors

We define our experiments to explore as much of the parameter space as is possible. For each parameter we limit our search to a subset of the defined range within which a measurable difference in network response is known to exist [2]. The adapted gossiping protocol takes six parameters, labelled X_1 to X_6 :

1) *Gossip probability*: The gossip probability X_1 defines the likelihood of packet rebroadcast upon receipt by a non-destination node. This is unitless and defined in the range $[0, 1]$. Search range is $[0, 1]$.

2) *Seen-packet buffer*: The seen-packet buffer size X_2 controls the number of previously seen packets the node will store. This is measured in *packets*, and is defined in the range $[0, \infty)$ for integral values only. Search range is $[1, 10]$.

3) *Waiting-packet buffer*: The waiting-packet buffer size X_3 controls the number of packets stored in a queue to be broadcast. This is measured in *packets*, and is defined in the range $[1, \infty)$ for integral values only. Search range is $[1, 10]$.

4) *Initial backoff*: Just prior to transmitting a packet, nodes test the wireless medium to ensure no other transmissions are occurring which would interfere with the proposed broadcast. If another transmission is detected the node waits for a backoff period, X_4 , and each subsequent failed attempt waits for the n th power of this backoff value as X_4^n . This is measured in *seconds*, and is defined in the range $[0, \infty)$. Search range is $[0.1, 1]$.

5) *Packet lifetime*: The packet lifetime X_5 determines the length of time a packet is allowed to exist in the network before being dropped. Once it exceeds this value, all nodes drop the packet. This is measured in *seconds*, and is defined in the range $[0, \infty)$. Search range is $[0.1, 10]$.

6) *Intercluster TTL*: The intercluster TTL X_6 represents the maximum number of node-node hops a packet may make between clusterheads. If the number of hops exceeds this before reaching the destination, the packet is dropped. This is measured in *hops*, and is defined in the range $[1, \infty)$ for integral values only. Search range is $[1, 10]$.

E. Output metrics

In order to compare network performance, a set of metrics must be defined over which networks can be measured. Tate *et al.* [2] define several metrics addressing the performance, reliability, and efficiency of the network given the parameter tunings applied to it. Reliability is measured by the packet delivery failure ratio. Performance is measured by the average latency per packet, and can be measured per hop and per metre traveled. Similarly, efficiency is measured as the average energy required to send a packet one hop or one metre. For all metrics, lower values indicate a higher quality solution. A value of zero represents a perfect solution in a given metric, although in practice this may not be attainable; the optimal value may be somewhat higher, whether or not this optimal value is known. Metrics are labelled M_1 to M_5 . Definitions of these are given in [2].

1) *Performance*: Network latency is the average time for packets to traverse unit distance in the network. M_1 defines unit distance in terms of physical distance in metres, measured in *seconds per metre*. M_2 defines unit distance in terms of logical distance in node-to-node hops, measured in *seconds per hop*. M_1 and M_2 are defined in the range $(0, \infty)$.

2) *Reliability*: The packet delivery failure ratio metric M_3 represents the percentage of packets created by the simulated application which the network attempted to deliver but were lost before reaching the intended destination. M_3 is unitless and defined in the range $[0, 1]$.

3) *Efficiency*: The energy metrics represents the average energy required for a unit packet to traverse the network by a unit distance. If the unit distance is defined in terms of physical distance in metres, the resulting metric M_4 is measured in *Joules per packet per metre*. If the unit distance is defined in terms of logical distance in node-to-node hops, the resulting metric M_5 is measured in *Joules per packet per hop*. M_4 and M_5 are defined in the range $(0, \infty)$.

F. Measuring solution quality

The metrics M_1 to M_5 defined above are all mutually independent and may be targeted as individual objectives by sensornet designers. However, real sensornet designs are likely to require an acceptable compromise between multiple competing objectives. It is therefore necessary to define a mechanism by which the relative quality of two or more candidate solutions can be compared to determine which offers the best compromise.

Assume we have n controlled factors X_1 - X_n and m metrics M_1 - M_m . A candidate solution $S_\alpha = \{X_{\alpha 1}, \dots, X_{\alpha n}\}$ maps to a set of metrics $T_\alpha = \{M_{\alpha 1}, \dots, M_{\alpha m}\}$. The mapping of $S \mapsto T$ is not known *a priori* but instead is evaluated experimentally as described in section III-B for specific values of S . A perfect solution $S_{perfect}$ would yield a set of metrics $T_{perfect}$ such that $\forall M \in T_{perfect} \bullet M = 0$. Although $S_{perfect}$ does not necessarily exist, we define the quality measure E in Equation 1 of any given candidate solution S_α based on the Euclidean distance from the point in solution phase space defined by T_α to the point $T_{perfect}$.

$$E = \sqrt[2]{\sum_{i=1}^m w_i (s_i M_i)^2} \quad (1)$$

Some network performance attributes may be of greater importance than others to a sensornet designer. We therefore define weighting w_i for metric M_i such that a larger weighting value indicates a greater importance attached to the network performance attributes quantified by a given metric.

Each of the metrics M_1 - M_m may be defined over a different range, so it is inappropriate to compare the absolute measured values directly. We define a scaling factor s_i for metric M_i such that all possible values of $s_i M_i$ are found in the range $[0, 1]$, noting that the ideal value of any given metric is also the lowest possible value, 0. It is only meaningful to compare two E values if all scaling values s_i are equal for each E . If for a given metric M_i is defined over a finite range then the value of s_i is well-defined and does not vary between network configurations under consideration. However, if a given metric M_i is defined over an infinite range then there does not exist a single well-defined value of s_i . Instead, we define s_i in the context of a given set of experimental results by setting $s_i = \frac{1}{MAX(M_i)}$ where $MAX(M_i)$ is the largest value of metric M_i observed.

In the experimental work that follows we set all $w_i = 1$ to give equal weighting to all metrics, and set all s_i using the second definition above as some metrics defined in section III-E are defined over an infinite range. It follows that all values of E are defined in the range $[0, 1]$ where 0 is the solution quality deriving from the theoretically perfect solution and 1 is the solution quality deriving from the worst quality solution considered in the set of all experiments.

IV. DOE APPROACH: FACTORIAL DESIGN

In this section we define the experiments with which the parameter landscape is explored, at broad scope but shallow depth. Factorial design methods are used to define the experiment set, and linear interaction model fitting methods are used to analyse the results.

A. Two-phase experiment design

Full factorial design [6] is used to systematically explore the entire parameter landscape. This approach gives broad but shallow coverage of all possible combinations of all acceptable ranges of controlled factors. Statistical models are fitted to experimental results to yield a generalised model of the relationship between controlled factors and each measured response. This model is useful for predicting likely network performance for any arbitrary set of input values. The model can be used in the opposite direction by defining sections of the multi-response hypersurface corresponding to the desired network performance, and working backward to input values by solving the simultaneous equations of the fitted model to yield a set of inequalities defining usable ranges of input controlled factors.

We address the combinatorial explosion by applying a two-phase method. Phase 1 allows the experimenter to identify

which of the controllable factors are actually important, and which can be safely ignored. Phase 2 explores the significant controllable factors in much greater detail. The experimenter can therefore avoid wasting resources and analytical effort on matters which will not significantly influence the outcome, and more detailed statistical models can be derived for the same experimental cost.

In Phase 1 we identify which of the protocol controlled factors are the best predictors of the network performance metrics. This requires a small number of points in the parameter space to be sampled in the axis corresponding to each controlled factor, and a set of simulation experiments to be run to measure network performance under each combination. The ANOVA method is applied to assess which controlled factors are significant to the experimental outcomes [10]. Any factors which are deemed statistically insignificant are dropped at this stage.

In Phase 2 we sample the parameter space along the corresponding axis in a greater number of points for each statistically significant controlled factor. Again, a set of simulation experiments was performed to measure network performance under each configuration. Statistical models are then fitted to the output metrics resulting from these experiments, and these models are then used to predict the best set of values to assign to controllable factors. Phase 1 identifies which controllable factors are not significant to the outcome, so the specific value we assign to each of these insignificant factors is unimportant, provided that the selected value falls within the boundaries explored in Phase 1. We select the midpoint value of the boundaries from section III-D for each controllable factor deemed insignificant.

B. Cost analysis

Given p controlled factors, each sampled at q points in the permitted region, we have q^p protocol configurations to assess. We assess each protocol configuration with r networks to prevent results being unduly influenced by a given network design, yielding $r q^p$ experimental configurations to consider by simulation. We repeat each experimental configuration s times to prevent results being unduly influenced by any single unusual simulation instance, yielding the requirement to run $r s q^p$ simulations in total.

Assuming each simulation completes in approximately equal wall time, t , we find that total experiment time grows exponentially in p , polynomially in q , and linearly in r and s . As total experiment time is NP-hard in p it is obvious that any reduction in p is valuable, and is more significant than similar reductions in q , r or s . Phase 1 addresses this problem by identifying controllable factors which can safely be disregarded. It is therefore possible in Phase 2 to increase q after reducing p and still have the full experiment set complete in acceptable wall time.

All simulations are mutually independent and can therefore be executed in parallel, reducing total runtime to that of a single simulation if sufficient processing hosts are available. Assume a multiprocessing environment in which $x \in \mathbb{N}$ independent simulations can execute in parallel. For DOE

experiments there are no dependencies between simulations so any number can execute in parallel, all at cost t . The total wall time cost is $C = \frac{rsq^p}{x}t$. Note that $C \propto \frac{1}{x}$, reaching a minimum of $C = t$ where $x = rsq^p$.

Increasing the number of experimental configurations increases the quality of fitted statistical models, and hence solution quality, but also increases experiment cost. A balance must be found which obtains solutions of acceptable quality within reasonable time. We measured wall time for all experiment simulations and took the mean as $t = 78.51s$.

In both Phases 1 and 2 we set $r = 3$ and $s = 3$. In Phase 1 we set $p = 6$ and $q = 3$, yielding a requirement for $3 \times 3 \times 3^6 = 6561$ independent simulations. As we will fit linear models to the results in section IV-C we must consider at least two values for each controlled factor, but to improve accuracy we use three. we take one value at the low extreme of the defined interval, another value at the high extreme, and another value from the centre. This ensures that the results cover the full spectrum of possible behaviour. In Phase 2 we set $p = 3$ and $q = 6$, yielding a requirement for $3 \times 3 \times 6^3 = 1944$ independent simulations. As the number of controlled factors has decreased it is feasible to use more values for each, yielding a more detailed model.

C. Model fitting

The factorial design of the experiment suite described in section IV-A samples the parameter space at q^p points as described in section IV-B. These pairs of sample points and simulation-derived metrics represent exact solutions to specific known points in the generalised model of the relationship between controlled factors and output metrics. However, these are not directly usable if we wish to know the relationship between input and output, or vice-versa, for other points in the input-output phase space.

To consider points in the parameter space that have not measured directly we need to interpolate by fitting a statistical model to the known sampled points to derive a set of equations describing a hypersurface in the phase space [10]. We then work with the fitted surface rather than specific individual experimental results. An appropriate statistical model must be selected, which yields a surface with shape similar to that which would be observed if an infinite number of sample points were used. Previous work [2] has shown that linear first-order interaction models are a suitable approximation in the context of the TTL-bounded gossip algorithm in sensornets.

Sampling the parameter space at more points yields a fitted model which is a better approximation of the real relationship by providing more data for the model fitting algorithm. For a finite set of sample points there exists the risk that an interesting feature of the solution landscape falls between sample points, and hence is not present in the fitted model. The DOE broad-but-shallow search implemented by factorial design experiments may or may not outperform an Evolutionary Algorithm in this regard; interpolation allows every candidate parameter set to be considered simultaneously, including those not measured directly, but there is a

risk that the optimal solution lies between directly measured points and is not revealed in the fitted model.

For each output metric under consideration, a linear interaction model of the form given in Equation 2 was fitted to the result set in MATLAB. β_0 is a constant, X_i is the i th controlled factor value, β_i is the coefficient for controlled factor X_i , β_{ij} is the coefficient for the interaction between controlled factors X_i and X_j , and ε is the normally-distributed noise term. The response Y is influenced linearly by each factor and each pairing of potentially interacting factors. Lack of space precludes the controlled factor coefficients extracted from Phases 1 and 2 being included in this paper; these can be found at [11].

$$Y = \beta_0 + \sum_{i=1}^n \beta_i X_i + \sum_{i=1}^n \sum_{j=i+1}^n \beta_{ij} X_i X_j + \varepsilon \quad (2)$$

For each output metric M_1 - M_5 a separate linear interaction model is produced in which 6 axes represents controlled factors X_1 - X_6 and a further axis in which the height of the hypersurface varies with the values of the output metric M_i . Axes corresponding to controlled factors X_1 - X_6 are common to all metrics M_1 - M_5 so a more complex surface can represent the interrelationships between all controlled factors and all metrics.

Finding sets of values for controlled factors corresponding to solutions with appropriate characteristics is equivalent to identifying regions of the axes representing controlled factors X_1 - X_6 with appropriate fitted surface height in the axes corresponding to output metrics M_1 - M_5 . Similarly, finding optimal or worst-cast sets of controlled factors is equivalent to finding minima and maxima in the fitted surface. This is implemented by solving sets of simultaneous inequalities when identifying regions with suitable characteristics, or by solving sets of simultaneous equations when addressing optimal or worst-cast characteristics.

D. Experimental results

Lack of space precludes the inclusion of all experimental results, even in summarised form; for the full results refer to [11]. However, it is important to consider the n -way ANOVA [10] results for the first-order pairwise interactions between these controllable factors, shown in Table I.

The results from Phase 1 [11] show that some controlled factors and factor interaction pairs are more significant than others, and the measure of significance R^2 for any given factor or factor pair tends to vary between metrics. However, it is evident that the controlled factors $\{X_1, X_5, X_6\}$ are significant in isolation with 95% confidence ($R^2 < 0.05$) for at least two of the metrics M_1 - M_5 , and at least one of $\{X_1, X_5, X_6\}$ is evident in almost all interaction pairs deemed significant with 95% confidence. Factors $\{X_2, X_3, X_4\}$ are not significant in isolation for any metric, or as a member of an interaction pair which does not include any of $\{X_1, X_5, X_6\}$. We therefore select controlled factors $\{X_1, X_5, X_6\}$ for Phase 2 and discard factors $\{X_2, X_3, X_4\}$.

Table I presents similar data corresponding to the subset of controllable factors deemed statistically significant and explored in greater detail in Phase 2. R^2 values are defined in the interval $[0, 1]$ where lower values indicate greater statistical significance. All figures are presented to 4 decimal places; particularly small R^2 values appear rounded to 0.0000 but are non-zero positive members of \mathbb{R} .

	M_1	M_2	M_3	M_4	M_5
X_1	0.1034	0.0000	0.0000	0.0000	0.0000
X_5	0.1417	0.1546	0.0000	0.0000	0.0001
X_6	0.0000	0.0000	0.0000	0.0000	0.0000
$X_1 \times X_5$	0.6780	0.4134	0.9365	0.7561	0.5617
$X_1 \times X_6$	0.7624	0.0652	0.0000	0.2421	0.5512
$X_5 \times X_6$	0.6058	0.9437	0.0000	0.0008	0.0381

TABLE I
PHASE 2: R^2 VALUES FOR CONTROLLED FACTORS $\{X_1, X_5, X_6\}$ AND THEIR INTERACTIONS FOR METRICS M_1 - M_5

Table I shows that the most significant factors from Phase 1 remain significant when considered in greater detail in Phase 2. Each of the factors $\{X_1, X_5, X_6\}$ is significant with confidence $> 99\%$ for each of metrics M_3 - M_5 , and each of these factors is significant with confidence of $> 85\%$ for the remaining metrics M_1 and M_2 . Both M_1 and M_2 are network performance metrics (see section III-E); we conclude that the model is good at predicting all metrics M_1 - M_5 , but is better at predicting network reliability and efficiency than performance.

Turning to the factor interactions, we observe that the $X_1 \times X_5$ interaction is largely irrelevant. The $X_1 \times X_6$ interaction is significant with $> 93\%$ confidence for metrics M_2 and M_3 . The $X_5 \times X_6$ interaction is significant with $> 94\%$ confidence for metrics M_3 - M_5 . We conclude that factor interactions are important in sensornet protocols and must be taken into account by sensornet designers.

The coefficients $\{X_1, X_5, X_6\}$ were inserted into the generalised form of Equation 2 given in section IV-C yielding a set of simultaneous equations. We solved these simultaneous equations to minimise the E measure described in section III-F, yielding a set of controlled factor values $\{X_1, X_5, X_6\}$ expected to approximate the optimal solution. For factors $\{X_2, X_3, X_4\}$, which are not statistically significant and are therefore omitted from the simultaneous equations, we take the midpoints of ranges defined in section III-D. The actual network performance metrics M_1 - M_5 corresponding to this calculated approximation to the optimal solution are given in section VI.

V. EVOLUTIONARY APPROACH: TWO-ARCHIVE

In this section we define the experiments with which the parameter landscape is explored, at narrow scope but substantial depth. The Two-Archive evolutionary algorithm is employed to progressively improve solution quality.

A. The Two-Archive algorithm

The Two-Archive algorithm is a multi-objective evolutionary algorithm developed by Praditwong and Yao [5]. It is designed to perform well on problems with a large

number of objectives. Two-Archive uses an *elitist* approach, storing the best solutions it has found at any given point in an archive. Unlike traditional archiving algorithms, Two-Archive separates this archive further, as described below.

One of the issues with archiving algorithms, particularly in high-dimensional problems, is the potential for memory overflow. To counteract this, archives must be limited to a certain size and reduced as necessary when this size is reached. However, non-dominated solutions such as those stored in the archives can be subdivided into two types: convergent solutions, which have dominated a previous member of the archive, and divergent solutions, which are merely non-dominated by all current members. In general, convergent solutions tend to help convergence towards the Pareto-optimal front, whereas divergent solutions may or may not lead towards a local optimal front instead. Whereas previous MOEAs do not differentiate between these solution types, Two-Archive stores each kind of solution separately.

When saving solutions to the archives, if the given solution has successfully dominated an existing member (or members) of either archive, it is placed in the convergence archive, and the dominated members of the archives are deleted. Otherwise, the given solution is added to the diversity archive, and no members are deleted. From this it can be seen that solutions entering the convergence archive do not increase the size of the combined archive, as at least one member must be dominated (and thus deleted) before it can enter the convergence archive. When entering the diversity archive, no members are deleted, and so the total archive size is increased by one. This is where the potential for memory overflow exists, and so Two-Archive implements a removal strategy, as given below.

When the total archive limit is exceeded, the removal strategy is executed to decrease the size of the archives. During this process, only members from the diversity archive are removed, ensuring convergent solutions remain to encourage convergence to the Pareto optimal front. The removal strategy works by calculating the shortest Euclidean distance from each member of the diversity archive to any member of the convergence archive. Solutions with the least distance to the convergence archive members are then removed until the archive size is within acceptable limits. As a result of this strategy, the size of the archives during runtime is never more than twice the population size, and the final archive size will never be greater than the population size.

Selection of a new mating population also comes from the archives. This is accomplished by selecting an archive according to a specified probability, and then randomly selecting a solution from that archive. This process is then repeated until the mating population is full.

Two-Archive operates as follows. First, a random initial population is generated and evaluated according to the given fitness function. The algorithm then enters a loop. Non-dominated solutions in this population are added to the archives as described, with the removal strategy applied at the end if the archives overflow. The mating population is

then constructed as detailed above, and genetic operators are applied to this population to produce the next generation of solutions. These are then evaluated, and the loop repeats until a termination condition is reached. The contents of the archives are then taken as the final population.

Two-Archive exhibits convergence similar to that of the other leading MOEAs, yielding results of similar or higher quality. Experimental work indicates that Two-Archive run-time for a given problem is typically equal or lower than that displayed by these other algorithms, by up to an order of magnitude for representative problems [5]. This does come at a cost of reduced diversity in comparison to some alternate MOEAs. However, we assert that suboptimal diversity is tolerable for the tuning experiments considered in this paper; the losses in diversity are more than offset by the gains in efficiency which enable a significantly greater number of generations to be evaluated per unit time.

B. Two-Archive experimental settings

A parameter, r , defines the ratio of parent selection from the convergence and diversity archives. A higher ratio leads to more convergence solutions being chosen, and thus faster convergence overall at the cost of potentially reduced diversity in the range of solutions. Population size, s , defines the size of the combined archives. We set $r = 0.9$ and $s = 50$.

Simulated Binary Crossover (SBX) [12], polynomial mutation [12] and random selection operators were used for this experiment. SBX takes as parameters a crossover rate, c and an η_c value controlling the probability of *near-parent solutions* being generated (with higher values producing closer matches to parents). These were set as 0.7 and 15, respectively, for this experiment. Polynomial mutation also takes a mutation rate, m , and an η_m value controlling the *mutation distance*, which were set to 1/6 and 20 in this case.

Experiments were conducted in which all values were represented internally as 64-bit precision floats. Where a given parameter is defined only for integral values, the float value was rounded to the nearest integer at the point of use. All tests were run for 50 generations, to give the solutions time to converge to useful values. Data on the best known candidate solutions were logged at every generation to provide insight into the running convergence of the system.

C. Cost analysis

Each fitness function evaluation requires exactly one simulation instance to be executed. Consider an evolutionary run with a population size of a for which b generations are required to attain the required solution quality. Within each generation it is necessary to evaluate the fitness function once for each candidate solution, requiring ab simulation instances for all population members and all generations. As with the DOE experiments, we test r networks and repeat each experimental configuration s times, requiring $abrs$ simulations in total. Total cost grows linearly in each of a , b , r and s ; this is clearly a desirable property.

We now consider the relative costs of the EA approach as described above, and the DOE approach described in

section IV-B. Assume each simulation instance completes in t seconds. The DOE approach has wall time cost $C_A = rsq^p t$ and the EA approach has wall time cost $C_B = abrst$. Given a single uniprocessor host, the EA approach will terminate before the DOE approach if $C_B < C_A$, a condition which is fulfilled where $ab < q^p$.

Now assume a multiprocessing environment in which x independent simulations can execute in parallel. For DOE experiments there are no dependencies between simulations so any number can execute in parallel, all at cost t . The total wall time cost is $C_C = \frac{rsq^p}{x} t$. Note that $C_C \propto \frac{1}{x}$, reaching a minimum of $C_C = t$ where $x \geq rsq^p$. For EA experiments it is possible to run all ars simulations of a given generation in parallel at cost $arst$, but all simulations of a given generation must complete before the next generation can begin. The total wall time cost is $C_D = \frac{ars}{x} bt$. Note that $C_D \propto \frac{1}{x}$, reaching a minimum of $C_D = bt$ where $x \geq ars$. If x is large then DOE experiments will complete before EA experiments.

VI. RESULTS

Tables II and III summarise the output of the DOE experiments described in section IV and the EA experiments described in section V. We label the Design Of Experiments approach as A and the evolutionary approach as B . Sets of protocol tuning values corresponding to A and B are labelled I_A and I_B respectively. Figures are to 4 decimal places.

For each experimental approach, the set of values assigned to controlled factors X_1 - X_6 corresponding to the highest quality solution discovered is given in table II. For approach A some controlled factors were not evaluated directly in Phase 2 of the experiment. For these controlled factors, italicised in table II, we take the midpoint of search ranges defined in section III-D.

	X_1	X_2	X_3	X_4	X_5	X_6
I_A	0.9999	<i>5.5000</i>	<i>5.5000</i>	0.5500	4.9506	7.5390
I_B	0.8425	5.0000	3.0000	0.3864	6.8320	9.0000

TABLE II
BEST-KNOWN PROTOCOL TUNINGS

We define the highest quality solution I_α for approach α as being that which offers the smallest Euclidean distance E_α between O_α and the theoretical perfect values of metrics, as defined in sections III-E and III-F. Table II shows the Euclidean distances E_A and E_B from which I_A and I_B were identified as the highest quality solutions derived by approaches A and B respectively. Note that the theoretical perfect metric values are not necessarily attainable under *any* real protocol tuning.

To ensure fair comparison of the quality of solutions obtained by the two experimental approaches, it is necessary to eliminate any factors which could unfairly influence the outcome. We achieved this goal by conducting further simulation experiments as per section III where the simulation scenario is identical in all respects, except for the protocol parameter set which is either I_A or I_B as appropriate.

Three hundred simulations were executed for each of I_A and I_B as defined in table II; 100 repeats for each of the

3 networks considered in the experiments of sections IV and V. Where a controlled factor X_1 - X_6 is defined only for integral values, but the value given in table II is non-integral, we round to the nearest integer. For each combination of experimental approach and metric M_1 - M_5 , a set of 300 output values is produced. The arithmetic mean of each set is taken as the final value and presented in table III. The sets of output metrics corresponding to A and B are labelled O_A and O_B respectively. $f(O_B, O_A)$ gives O_B as proportion of O_A to allow comparison of relative solution quality. All figures for M_1 - M_5 are given to 5 significant figures and scaled by a factor of 10^6 for clarity.

	M_1	M_2	M_3	M_4	M_5	E
O_A	14624	7.2833	352420	75106	36.026	0.18771
O_B	15192	8.0257	368900	5948.3	2.9428	0.20370
$f(O_B, O_A)$	1.0388	1.1019	1.0468	0.0792	0.0817	1.0852
Best	DOE	DOE	DOE	EA	EA	

TABLE III

NETWORK PERFORMANCE FOR BEST-KNOWN PROTOCOL TUNINGS

It is notable the approaches find different result sets, as shown in Table II. The observed metrics as shown in Table III are also different, with DOE producing better results in terms of network performance (X_1 and X_2) and reliability (X_3), but the EA approach producing results with significantly better energy efficiency (X_4 and X_5). It is noteworthy that for ($X_1 - X_3$) both DOE and EA approaches yield results of the same order of magnitude, but for (X_4 and X_5) the EA yields results that are superior by an order of magnitude.

The DOE approach yields a compromise solution that is superior to the compromise solution found by the EA approach, as measured by the Euclidean distance metric E . However, in problems with larger sets of input variables, section V-C demonstrates that the DOE approach will become prohibitively expensive in comparison to the EA technique, although with smaller sets of inputs the DOE method may be faster. Similarly, when a simple ranking metric such as Euclidean distance is unavailable or unsuitable, the EA approach will provide a better diversity of possible solutions.

During the evolution of the EA results the solution quality was not observed to supplant that of the DOE results. Although theoretically possible, it is considered unlikely to occur within acceptable time. However, the EA results made good progress during the experimental runtime, rapidly approaching the observed optimum. Consequently, the EA approach can yield useful near-optimal results early in the evolutionary run, whereas the DOE approach yields no useful results before termination.

We designed our DOE and EA methods to find a single best solution to a protocol optimisation problem. As EA approaches are generally optimised for finding sets of non-dominated solutions rather than single solutions, experimenters may need to allocate more fitness function evaluations and experimental effort than required under DOE approaches to obtain single solutions of equivalent quality.

The DOE-derived fitted model is a useful summary of the relationship between inputs and outputs across the full

defined ranges of all inputs, including solutions correspond to poor performance, whereas the EA approach focuses computation resources on good solutions. We therefore conclude that both approaches serve useful though different purposes. The sensornet designer might usefully apply the DOE approach to narrow the search space to those portions of the parameter space mapping to useful portions of the solution space, then apply the EA approach to navigate any non-linear regions within this narrowed search space to obtain better near-optimal solutions.

VII. CONCLUSIONS

In this section our findings are considered against the research objectives defined in the introduction. For the first objective, *Obj 1*, the results obtained from the DOE and EA approaches shown in Tables II and III show that both methods achieve results close to the theoretical optimum for this problem. For the second objective, *Obj 2*, the metrics presented in Table III show that DOE outperforms the EA approach in network performance, reliability, and overall solution quality metrics. However, the EA approach produces reasonable results across all metrics, and outperforms the DOE approach in energy efficiency metrics.

ACKNOWLEDGEMENT

This work is partially supported by an EPSRC grant (No. EP/D052785/1) on "SEBASE: Software Engineering By Automated Search".

REFERENCES

- [1] D. Estrin, L. Girod, G. Pottie, and M. Srivastava, "Instrumenting the world with wireless sensor networks," in *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, vol. 4, 2001, pp. 2033–2036.
- [2] J. Tate, I. Bate, and S. Poulding, "Tuning protocols to improve the energy efficiency of sensornets," in *Proceedings of the Fourth UK Embedded Forum*, 2008.
- [3] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II," in *Proceedings of the 6th International Conference on Parallel Problem Solving from Nature*, 2000, pp. 849–858.
- [4] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the Strength Pareto Evolutionary Algorithm," in *Proceedings of the EU-ROGEN2001 Conference*, 2001, pp. 95–100.
- [5] K. Praditwong and X. Yao, "A new multi-objective evolutionary optimisation algorithm: The Two-Archive Algorithm," in *Proceedings of the International Conference on Computational Intelligence and Security*, 2006, pp. 286–291.
- [6] A. Baker, S. Dixon, F. Drabble, J. Gibbings, A. Lewkowicz, D. Moffat, and R. Shaw, *The Systematic Experiment*, J. Gibbings, Ed. Cambridge: Cambridge University Press, 1986.
- [7] J. Tate and I. Bate, "YASS: A scalable sensornet simulator for large scale experimentation," in *Proceedings of Communicating Process Architectures*, 2008, pp. 411–430.
- [8] T. He, B. Blum, Q. Cao, J. Stankovic, S. Son, and T. Abdelzaher, "Robust and timely communication over highly dynamic sensor networks," *Real-Time Systems*, vol. 37, no. 3, pp. 261–289, 2007.
- [9] S. Ni, Y. Tseng, Y. Chen, and J. Sheu, "The broadcast storm problem in a mobile ad hoc network," in *Proceedings of ACM/IEEE MobiCom*, 1999, pp. 151–162.
- [10] G. Box, J. Hunter, and W. Hunter, *Statistics for Experimenters: Design, Innovation, and Discovery*, 2nd ed. Wiley-Interscience, 2005.
- [11] "Supplementary experimental results for linear interaction models," <http://www.cs.york.ac.uk/~jt/content/nov08ic.pdf>, November 2008.
- [12] K. Deb and R. B. Agrawal, "Simulated binary crossover for continuous search space," *Complex Systems*, vol. 9, pp. 115–148, 1995.