A Case Study in Meta-Simulation Design and Performance Analysis for Large-Scale Networks

David Bauer, Garrett Yaun, Christopher D. Carothers
Murat Yuksel and Shivkumar Kalyanaraman

Department of Computer Science
Rensselaer Polytechnic Institute
Troy, NY 12180, U.S.A.
{bauerd,yaung,chrisc}@cs.rpi.edu, {yuksem,shivkuma}@ecse.rpi.edu

April 12, 2004

Abstract

Simulation and Emulation techniques are fundamental to aid the process of large-scale protocol design and network operations. However, the results from these techniques are often view with a great deal of skepticism from the networking community. Criticisms come in two flavors: (i) the study presents isolated and potentially random feature interactions, and (ii) the parameters used in the study may not be representative of real-world conditions. The first issue (random isolated results) can be addressed by large-scale experiment design techniques that extract maximum information and confidence from a minimum number of carefully designed experiments. Such techniques can be used to find "good" results fast to guide either incremental protocol design or operational parameter tuning. The second issue (representativeness) is more problematic and relates to formulating benchmarks that to the greatest possible extent characterize the structure of the system under study. In this paper, we explore both cases , i.e. large-scale experiment design and black-box optimization (i.e. large-dimensional parameter state space search) using a realistic network topology with bandwidth and delay metrics to analyze convergence of network route paths in the Open Shortest Path First (OSPFv2) protocol. **By using the Recursive Random Search (RRS) approach to design of experiments, we find: (i) that the number of simulation experiments is reduced by an order of magnitude when compared to full-factorial design approach, (ii) it allowed the elimination of unnecessary parameters, and (iii) it enabled the rapid understanding of key parameter interactions**. From this design of experiment approach, we were able to abstract away large portions of the OSPF model which resulted in a execution time improvement of 100 fold.

1 Introduction

Performance analysis techniques are fundamental to the process of protocol design and network operations [1, 2, 3]. The high-level motivation of these techniques is simple: to gain varying degrees of qualitative and quantitative understanding of the behavior of a system under-test. A number of specific lower-level objectives include: validation of protocol design and performance for a wide range of parameter values (parameter sensitivity), understanding of protocol stability and dynamics, and studying feature interactions between protocols. Broadly, we may summarize the objective as a quest for general invariant relationships between network parameters and protocol dynamics [1, 2, 4].

Systematic design-of-experiments [1, 5] is a well studied area of statistics and performance analysis offering guidance in this aspect. A survey of relevant papers in the networking field suggests that such systematic techniques (e.g.: factorial designs, large-scale search) have not been used in the protocol design process or network operations process except possibly by measurement specialists. This ad-hoc approach to organizing simulation or testbed experiments has worked when we design and examine a small number of features, network scenarios and parameter settings. However, this method is likely to be untenable as we design newer protocols that will rapidly be deployed on a large-scale, or have to deal with a combinatorial explosion of feature interactions in large operational inter-networks. This point has also been made in a related context by Floyd and Paxson

[2], where they pinpoint three reasons why it is hard to simulate large networks: *scale, heterogeneity and rapid change*. The need for scalable simulation and meta-simulation tools is implicit in Floyd [3]'s statement: *"...we can't simulate networks of that size (global Internet). And even if we could scale, we would not have the proper tools to interpret the results effectively..."*

Beyond mere scaling of simulation platforms, our next need is meta-simulation capabilities, i.e. large-scale experiment design. Statistical experiment design considers the system-under-test as a black-box that *transforms input parameters to output metrics*. The goal of experiment design is to *maximally characterize* the black-box with the *minimum number of experiments*. Another goal is *robust* characterization, i.e., one that is minimally affected by external sources of variability and uncontrollable parameters, and can be specified at a level of confidence. Beyond characterization, the methodology aims to *optimize* the system, i.e. allows one to find the appropriate input parameter vector that elicits the best output response. The underlying premise of experiment design is that each experiment (e.g.: a simulation run) has a non-negligible cost.

While regression models for small dimensional parameter spaces can be built using simple factorial methods [1, 5], these methods do not ramp up to large-scale situations. Usually as the size of a model is increased in either space or number of parameters, the modeler typically retreats to a sub-goal of characterization, and instead focus on optimization alone (a.k.a. black-box *optimization*). As a result, we replace detailed regression-like characterization with heuristic search methods. In this class of methods, a variety of "exploration" techniques are used to find regions of interest. For example, hill climbing is used to find the local extrema [6]. Many heuristic search algorithms have been proposed such as multi-start hill-climbing[7], genetic algorithms[8] and simulated annealing[9]. While these techniques tend toward the global optima in the limit, they do not have the property of finding good results quickly, i.e. they lack early-stage efficiency. We have recently proposed an efficient search algorithm (called Recursive Random Search [10]) for efficient large-dimensional heuristic parameter optimization. This approach thus far has yield very positive results in finding "good" global minima with few simulation runs.

The key focus here is a case study in the application of this meta-simulation technique to examine OSPFv2 convergence times during network link failures. This study includes OSPF optimizations for sub-second convergence, adapted from [11]. Here, *convergence* is defined to be time at which *all* routers in the network have a synchronized routing table or put another way, a consistent view of the routing tables is shared by all routers. We explore the cases , i.e. large-scale experiment design and black-box optimization (i.e. large-dimensional parameter state space search) using realistic topologies with bandwidth and delay metrics to analyze convergence of network route paths in the Open Shortest Path First (OSPFv2) protocol.

By using Recursive Random Search (RRS) approach to design of experiments, we find: (i) that the number of simulation experiments that must be run is reduced by an order of magnitude when compared to full-factorial design approach, (ii) it allowed the elimination of unnecessary parameters, and (iii) it enabled the rapid understanding of key parameter interactions. From this design of experiment approach, we were able to abstract away large portions of the OSPF model that result in a 100 fold improvement in simulation execution time.

In the next section we describe the term meta-simulation and it's relation to design of experiments. Then in Section 3, we present the OSPFv2 model, and the environment in which we generated our results. In Section 4, we explain how Recursive Random Search allows us to generate more detailed results with fewer experiments. In the final sections, we detail the results of our experiment designs and what we have learned from them.

2 Meta-Simulation: Large-Scale Experiment Design and Analysis

The purpose of the large-scale experiment design area of our research is to systematically formulate and organize multiple simulation experiments in the quest of the general invariant relationships between parameters and protocol performance response. To this end, we being the discussion with an overview of full-factorial design of experiments.

2.1 Overview of Full-Factorial Design of Experiments

Design of Experiments or "experiment design" is a well known branch of performance analysis, specifically, a sub branch of statistics [1, 5]. It has been used extensively in areas such as agriculture, industrial process design and quality control [5] , and has been introduced to the area of practical computer and network systems design by Jain [1]. Statistical experiment design views the system-under-test as a black-box that transforms input parameters to output metrics. The goal of experiment design is to maximally characterize (i.e. obtain maximum information about) the black-box with the minimum number of experiments. Another goal is robust characterization, i.e., one that is minimally affected by external sources of variability and uncontrollable parameters, and can be specified at a level of confidence.

The underlying premise of experiment design is that each experiment has a non-negligible cost. Simple designs like "best-guess" or "one-factor-at-a-time" designs are less favored in complex situations since they do not provide information about the

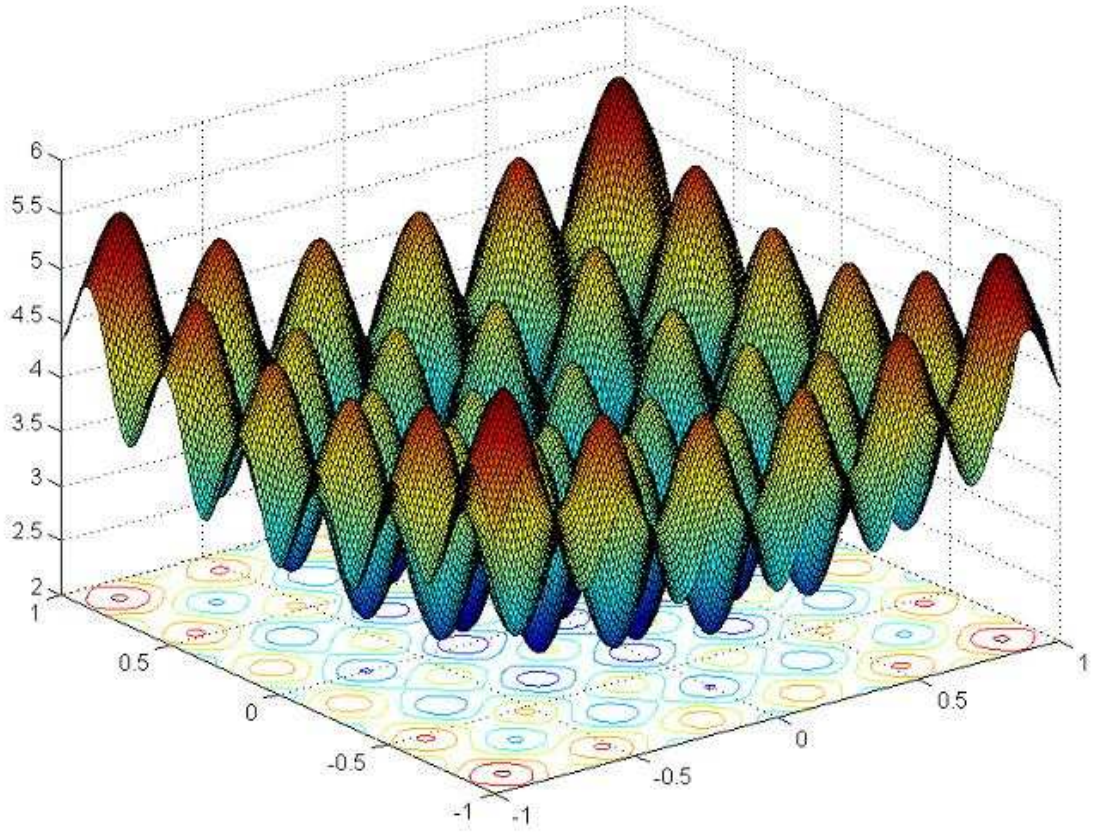

Figure 1: The benchmark Schwefel function as the response surface for a parameter space.

interactions between parameters. Designs like full-factorial and fractional factorial (also called orthogonal designs), appropriately subjected to replication, randomization and blocking are preferred [1, 5]. The usual end-goal of formulating regression models is to observe the effects of both individual parameters and parameter interactions [1, 5]. Techniques like blocking and analysis of covariance are used to explicitly handle measurable, but uncontrollable (a.k.a. "nuisance") factors. Transforms on data (e.g., Box-Cox power-law family of transformations) can effectively aid in producing a family of non-linear regression models and stabilizing the variance of the response [1, 5].

The next step beyond characterization (i.e. developing input-output regression models) is optimization, i.e. to determine the region in the important factors that leads to best-possible response. The output (i.e. response) in general will have an unknown surface topology, also known as "response surface" [1]. The approach typically used involves quickly traversing the surface sequentially (by using lower-order models built with fractional factorial experiments) to reach interesting areas where more detailed (higher-order) characterization is done.

As well known, one of the significant drawbacks of the full-factorial approach is the exponential in the number of experiments that must be run as a function of the number of data points per parameter. To vastly reduce the number, search algorithms must be used, such as RRS.

2.2 Overview of The RRS Algorithm

The key idea behind RRS is to maintain the initial efficiency of random sampling by "restarting" it before its efficiency becomes low. However, unlike the other methods, such as hill climbing, random sampling cannot be restarted by simply selecting a new

[1] An example response surface is shown in Figure 1

starting point. Instead we accomplish the "restart" of random sampling by *changing its sample space*. We perform random sampling for a number of times, then move or resize the sample space according to the previous samples and start another random sampling in the new sample space. Given a black-box objective function, a desired optimization process should start with inspecting macroscopic features of the objective function, and then look further into microscopic features in selected promising areas. The search process of RRS algorithm is fully consistent with this idea. In the beginning of the search, RRS performs sampling from the whole parameter space and thus examines the overall structure of the objective function. With the search continuing and the sample space gradually shrinking, the search gets more and more details of the objective function until it finally converges to a local optimum.

A stochastic search algorithm usually comprises two elements: *exploration* and *exploitation*. Exploration examines the macroscopic features of the objective function and aims to identify promising areas in the parameter space, while exploitation focuses on the microscopic features and attempts to exploit local information to improve the solution quickly. Various search techniques can be used for these two purposes. Since macroscopic features are hard to characterized, unbiased search techniques, such as random search and random walk, are often used for exploration. Some algorithms also try to build a simple model to characterize the macroscopic features of the objective function and perform exploration based on this model. However, to choose an appropriate model for a certain problem is very difficult and requires extensive *a priori* knowledge. Local search methods are the most commonly used techniques for exploitation, and hence exploitation is also called *local phase* in many literature. Accordingly, exploration is also known as *global phase*. Derivative-based local search methods, such as quasi-Newton method[12] and deepest descent[13], are very efficient for differentiable objective functions, however, they are not suitable for many practical problem because of its sensitivity to noise and restriction on the differentiability of objective functions[14]. Direct search methods, such as Nelder-Mead simplex method[15] and pattern search[16], do not exploit the derivative of objective functions and are more suitable for the concerned problems.

The RRS algorithm uses random sampling for exploration and recursive random sampling for exploitation. Ideally it should only execute the exploitation procedure in promising areas. However, it is difficult to determine which areas are more promising and should be exploited. Many algorithms, such as multistart, do not differentiate areas and hence may waste time in trivial areas.

For the algorithmic details and implementation of RRS we refer the interested reader to [28].

3 The OSPFv2 Design of Experiments

The goal of our design of experiments was to understand the factors determining the amount of wall-clock time required for a network of routers to detect and propagate a link state failure. Our experiment is a simulation of a network of Internet routers all operating the OSPFv2 protocol as described in [22]. For an example router network, we selected the AT&T network, as described by Rocketfuel data [23]. This network description was determined by using various network probing techniques (i.e, traceroute). The AT&T network is challenging because of it's size and complexity.

3.1 OSPFv2

OSPFv2 is a link-state routing protocol designed to be run internal to a single Autonomous System. Each OSPFv2 router maintains an identical database describing the internal network's topology (i.e. an Autonomous System (AS)). From this database, a routing table is calculated by constructing a shortest-path tree. OSPFv2 recalculates routes quickly in the face of topological changes, utilizing a minimum of routing protocol traffic. OSPFv2 is classified as an Interior Gateway Protocol (IGP). This means that it distributes routing information between routers belonging to a single Autonomous System. An example of an Autonomous System is the AT&T network, which is AS number 7018. Routing between ASs is handled by an external protocol, such as Border Gateway Protocol (BGP) [24].

The OSPFv2 protocol is based on link-state or shortest-path-first (SPF) technology. In a link-state routing protocol, each router maintains a database describing the Autonomous System's topology. This database is referred to as the link-state database. Each participating router has an identical database. Each individual piece of this database is a particular router's local state (e.g., the router's usable interfaces and reachable neighbors). The router distributes its local state throughout the Autonomous System via flooding. All routers run the exact same algorithm, in parallel. From the link-state database, each router constructs a tree of shortest paths with itself as the root. This shortest-path tree gives the route to each destination in the Autonomous System [22]. OSPFv2 routers employ the HELLO protocol for establishing and maintaining communications with adjacent routers. Adjacencies are established between two routers when a HELLO protocol packet is received by one of the two routers connected by a link. HELLO packets are then sent at regular intervals between adjacent routers. Upon receiving

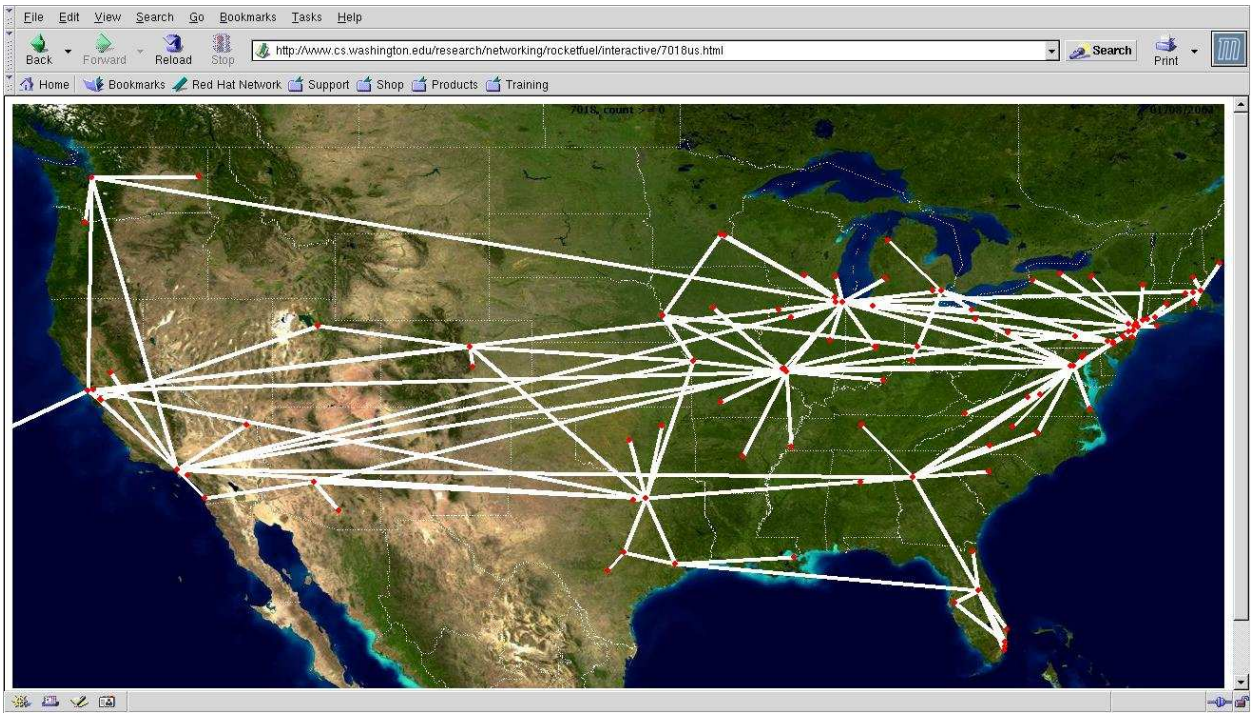

Figure 2: AT&T Network Topology from the Rocketfuel data bank.

a HELLO packet from a neighboring router, an inactivity timer is set for that router. If another HELLO packet is not received from that router before the timer expires, then the adjacency is broken and that router should no longer be used to route IP packets.

All of these aspects are modeled. However, multiple areas within a single OSPF domain is not currently modeled. In the experiments presented here, we configure OSPF to be a single large area. This was done because there is an interest in determining where OSPF ceases to execute in an efficient manner. This was a sub-goal of our experimentation.

3.2 AT&T Network Topology

For our network topology we selected the AT&T network. Figure 2 shows the core AT&T network topology which contains 11964 router nodes and 7491 links. Internet topologies like the AT&T network are interesting from a modeling prospective because of their sparseness and power-law structure [23]. This structure allows for a greater range of convergence times compared to fully connected networks. The OSPFv2 update packets require multiple hops in order to reach the outer edges of the network.

In performing a breadth-first-search of the AT&T topology, there are eight distinct levels. A number of routers were not directly reachable and thus were removed. Those routers are likely connected by transit routes. In total there are 3371 backbone routers and at the successive levels there are 8593 routers. The 4 ms delay that was chosen for the backbone core routers was in-line with the delays that Rocketfuel had associated with the Telstra topology backbone. An order of magnitude higher delay was selected for all lower level routers.

The bandwidth and delay for the AT&T topology is as follows:

- **Levels 0 and 1 routers:** 155 Mb/sec and 4 ms delay
- **Levels 2 and 3 routers:** 45 Mb/sec and 4 ms delay
- **Levels 4 and 5 routers:** 1.5 Mb/sec and 10 ms delay
- **Levels 6 and 7 routers:** 0.5 Mb/sec and 10 ms delay

Input Parameter	Minimum Value	Maximum Value
Hello Interval (seconds)	0.5	10.0
Hello Inactivity Interval (seconds)	1.5	8.0
ACK Timer Interval (seconds)	0.5	10.0
Maximum Transmission Unit (bytes)	500	1500
SPF Computation Interval (seconds)	0.5	10.0

Table 1: Random Recursive Search Input Plane.

Our experiments focused on the convergence time metric. We defined convergence to be the time at which all routers on the network have received an update corresponding to a link status change, and have recomputed their forwarding tables. In order to clearly state convergence intervals, in our simulations we have only a single link state failure per simulation and all of the OSPFv2 routers were started in a converged state. We defined the input plane to this experiment design to be composed of the HELLO interval, HELLO inactivity interval, SPF computation interval, ACK interval and maximum transmission unit. The response plane is the convergence time from the link state failure.

The goal for our design of experiments was to adapt some of the convergence optimizations in [11] for the IS-IS protocol to the OSPFv2 RFC [22] protocol. IS-IS is a link state protocol for Cisco routers. Suggestions for lowering convergence times were: to queue HELLO packets in front of data packets, use a modern shortest-path-first (SPF) algorithm, and to give a higher priority to link state packet (LSP) propagation over SPF computation. We took the following steps to adapt the optimizations.

We did not model the data plane in our OSPFv2 routers so that HELLO packets would always be at the front of the queue. It is still possible for other control plane packets to queue in front of the HELLO packets. To facilitate a higher priority for link state propagation over SPF computation, we remove the LSP propagation timer from the OSPFv2 protocol. Now, LSP propagation will always occur immediately, and the SPF computations will always occur later. In addition, modern SPF computations would only add a small amount of time to overall convergence interval. We modeled this by adding in the amount of time stated by [11] for a topology of our size.

4 Recursive Random Search Results

As previously discussed, Recursive Random Search (RRS) is a heuristic search algorithm for black-box optimization problems. This algorithm is specifically designed to optimize dynamic network protocol parameterizations with an emphasis on obtaining ”good” solutions within a limited time frame. RRS does not attempt to find a full optimization of the parameter space. The RRS algorithm maintains the high efficiency property of random sampling by constantly restarting random sampling but with adjusted parameter spaces. Because it shares this property with random sampling, it is also highly robust to the effect of random noises in the objective function. It also performs efficiently when handling an objective function that contains negligible parameters.

For the recursive random search algorithm we chose a wide range of input parameters, as shown in Table 1. We allowed RRS to search for 250 experiment runs, specifying a desired confidence level of 99%. RRS generated a convergence minimum after only 7 executions of 4.07 seconds. We fitted a linear regression model to our data using a tool called R [25], and generated the co-efficients shown in Table 2. After analyzing the variance on the inputs, we found the parameters that had the greatest impact on the model to be the HELLO packet interval, HELLO inactivity timer, and the SPF computation interval.

After considering the simulation model, we realized that the HELLO polling interval is set to be the HELLO packet interval multiplied by the HELLO inactivity interval. These two parameters are have an impact on convergence because they determine the time to detect a link state failure. The other factor of convergence time is the time to propagate the link state failure to the remainder of the routers in the network. The update propagation time is defined by flooding packets throughout the network. The router which detects the failure informs all remaining neighbors, who notify their neighbors, and so on, until eventually all routers in the network have received the update. The propagation delay on the updates is bounded by the amount of time it takes for the update to travel across the diameter of the network. Recall also from our definition of convergence that each router must have also recomputed their forwarding tables. So the convergence time is compounded by either how long it takes for the final router to update it’s table, or by the longest SPF computation interval.

After analyzing the variance we fitted a new linear regression model to the SPF computation interval and the cross between the HELLO packet interval and the HELLO inactivity interval, as shown in Table 4. This regression produced an adjusted R-squared value of 98%. In order to verify the accuracy and correctness in our model we created a scatter plot of the errors

Residuals

Min	1Q	Median	3Q	Max
-15.7286	-1.0311	0.1805	1.4811	11.5511

Coefficients

	Estimate	Std. Error	t value	$Pr(>\|t\|)$
(Intercept)	-19.838684	1.547959	-12.816	$< 2e-16$
HELLO Packet Interval	4.426648	0.121561	36.415	$< 2e-16$
HELLO Inactivity Interval	4.507337	0.169302	26.623	$< 2e-16$
ACK Interval	0.089194	0.113251	0.788	0.432
MTU	-0.001568	0.001146	-1.368	0.173
SPF Computation Interval	0.717926	0.121500	5.909	1.16e-08

Residual standard error: 4.17 on 243 degrees of freedom
Multiple R-Squared: 0.912, Adjusted R-squared: 0.9102
F-statistic: 503.5 on 5 and 243 DF, p-value: $<$ 2.2e-16

Table 2: RRS Linear Regression Model Output generated by R.

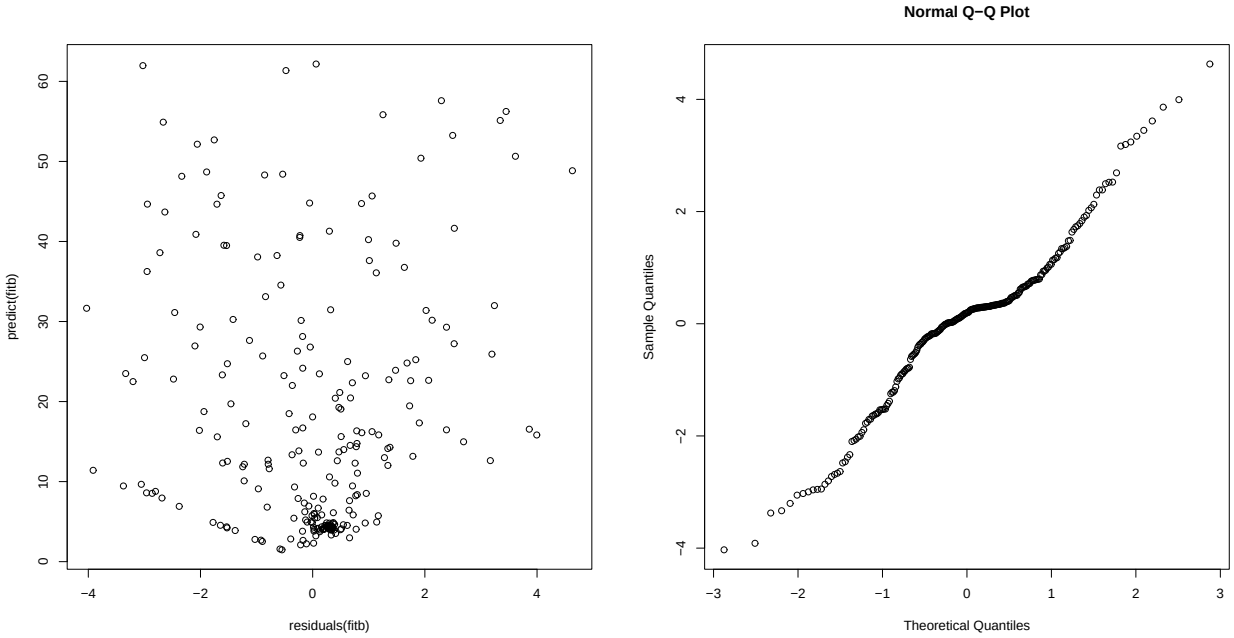

Figure 3: RRS Linear Regression Model Scatter and Q-Q Plots.

versus predicted responses. The scatter plot did not show any trends in the data. The next step in our verification was to create a quantile-quantile plot of the residual errors. We observe a linear relationship between sample error and theoretical. From Figure 3, the linear model assumptions of normality appear to be valid.

In order to gain more detail about the interesting parts of the design, we re-executed the model a second time with only

Input Parameter	Minimum Value	Maximum Value
HELLO Packet Interval (seconds)	0.5	10.0
HELLO Inactivity Timer (seconds)	1.5	8.0

Table 3: Re-parameterized Random Recursive Search Input Plane

Residuals

Min	1Q	Median	3Q	Max
-18.6508	-0.6173	0.23916	0.5768	5.3645

Coefficients

	Estimate	Std. Error	t value	$Pr(>\|t\|)$
(Intercept)	-0.31820	0.54756	-0.581	0.5617
HELLO packet Interval	-0.19646	0.11229	-1.750	0.0814
HELLO Inactivity Interval	0.13986	0.13361	1.047	0.2962
SPF Computation Interval	1.0743	0.2125	5.055	5.27e-06
HELLO interval × HELLO inactivity	0.92009	0.02436	37.773	$< 2e-16$

Residual standard error: 1.915 on 245 degrees of freedom
Multiple R-Squared: 0.9853, Adjusted R-squared: 0.9851
F-statistic: 5466 on 3 and 245 DF, p-value: $<$ 2.2e-16

Table 4: Re-parameterized Linear Regression Model Output generated by R

Input Parameter	Minimum Value	Maximum Value
HELLO Packet Interval (seconds)	0.03	1.0
HELLO Inactivity Interval (seconds)	1.5	2.0

Table 5: Re-scaled Random Recursive Search Input Plane

those input parameters. Specifically, we used only the HELLO packet interval, and the HELLO inactivity interval in the same ranges as shown in Table 3. Again, we allowed RRS to search for 250 iterations and with a confidence interval of 99%. This experiment generated a convergence minimum after only 129 executions of 0.93 seconds.

In this series of experiments, a sub-second range for the convergence interval is observed. Figure 4 shows that the HELLO packet interval and the HELLO inactivity interval form a plane with one corner tilting downward toward the smaller values. This low corner is anchored by the best convergence result given by RRS. We report a clustering effect occurring on the graph, which is attributed to the RRS algorithm centering upon a given input. It appears that RRS was successful in isolating the HELLO packet interval and the HELLO inactivity interval at the low end of their ranges.

The initial goal of our design was to determine if we could adapt some of Jacobsen's ideas to the OSPFv2 protocol and achieve convergence times on the order of magnitude in the tens of milli-seconds. Having isolated the effective parameters of the simulation model, we being to see points in the sub-second range. So we re-scaled the experiment into the range of the input values that generated those results. We noticed that the HELLO packet interval was in the range suggested by Jacobsen. We reduced the range for the input parameters as shown in Table 5 and Figure 5 shows the results of this experiment. All of the convergence values are below a second, and the best values are in range of tens of milli-seconds.

5 OSPF Model Critical Path Analysis

Berry and Jefferson [26], developed a technique called *Critical Path Analysis* to determine the optimal parallel simulation execution time. Armed with the results from RRS, we apply this technique here to examine what the critical path for OSPF

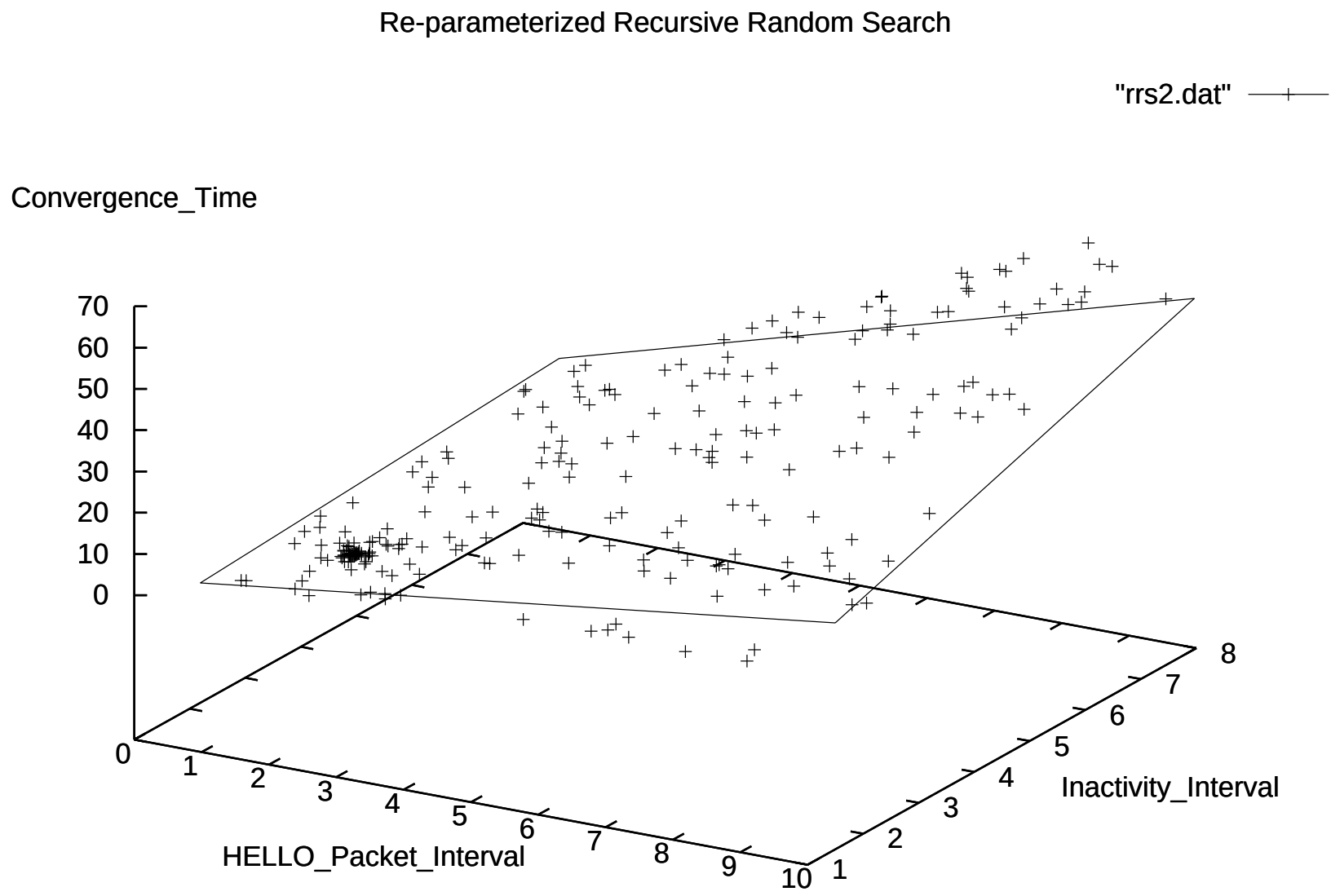

Figure 4: Re-parametrized Random Recursive Search.

convergence within the context of our model.

After examining the results from our designs, we observed that the two main components of convergence are *detection* and *propagation*. Detection is simply the amount of time that elapses between a link state change and the time at which the inactivity timer fires. The second component, propagation is determined by the longest path the link state update travels through the network. The longest path is not immediately determined by the number of hops between the originating router and the final router to receive the update. It is possible for an update to take many hops over high-speed links and still not be on the longest path. Conversely, it is possible to take only a small number of hops over very low speed links and be on the longest path. Realizing that these two factors have the highest impact on convergence time, **we observed that to accurately model convergence in any network, only the set of nodes which encompass the longest path through the network require simulation.** This observation has been used on other OSPF optimizations [27].

We simulated the AT&T network which contained almost 12,000 routers. The model was instrumented so that each router

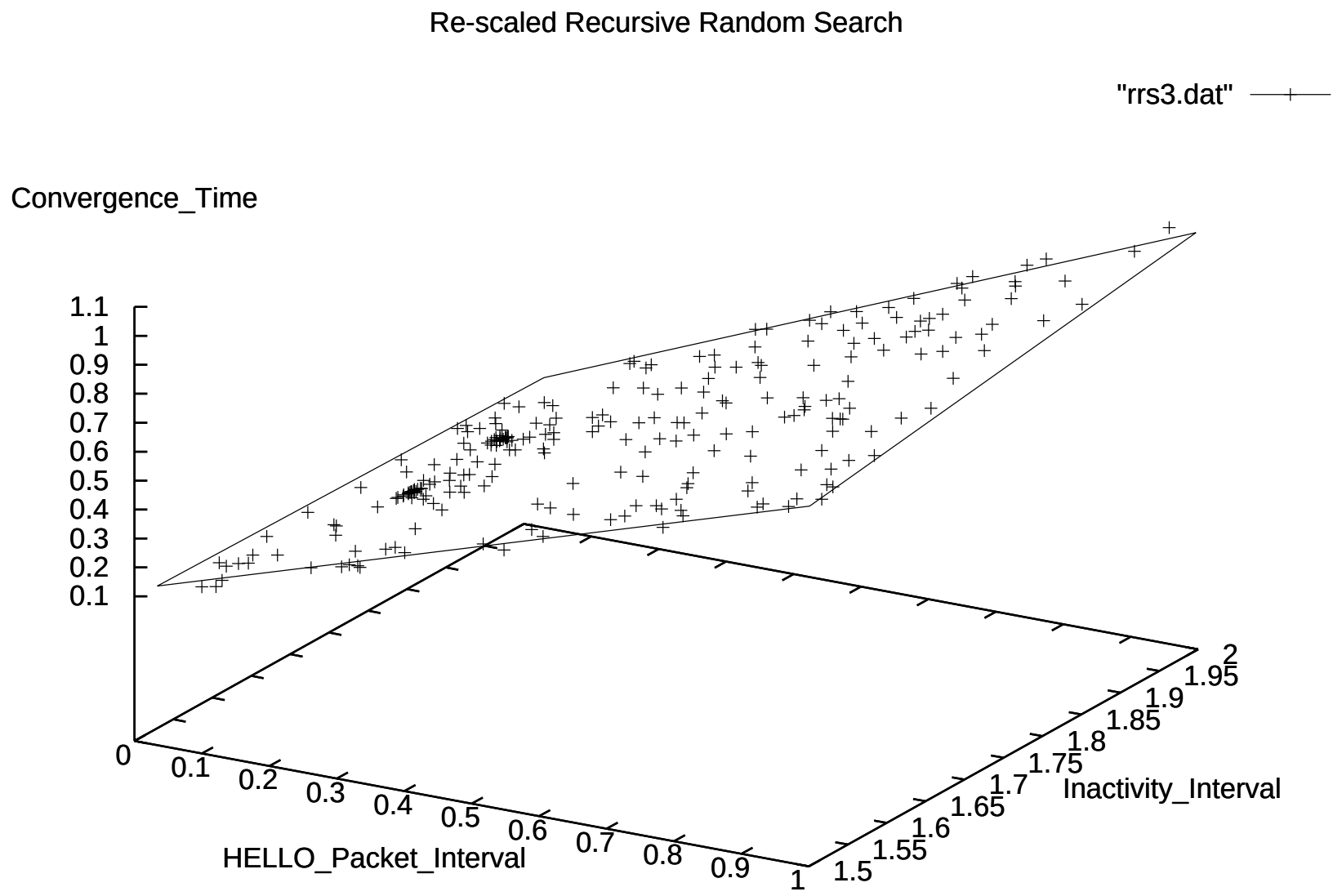

Figure 5: Re-scaled Random Recursive Search.

would keep track of which routers they received link state updates from. Once the update reached the final router in the network, we then backtracked this path to find the longest path in the network.

In order to validate the modeling optimization, we computed a full factorial design of the VSNL, India topology. This topology contained only 291 routers, which allowed us to compare the optimization to the full model simulation results. Table 5 shows the results of ten of those experiments runs. The optimization results generated exactly the same output for the optimization as we would have received had we modeled the entire topology.

Simulations on the full AT&T network required anywhere from one half hour to initialize to several hours depending on whether the routing tables need o be computed. In addition to the time required to initialize the simulation, execution time took on average one second of wall clock time to simulate one second of simulated time. In other words, to simulate an hour of OSPFv2 traffic required almost one hour of real time.

After determining this optimization, we computed the longest paths through the AT&T network for each of the updates

Experiment	HELLO Interval	HELLO Inactivity	ACK	MTU	SPF	Non-optimized	Optimized
1	0.5	1.5	0.5	500	0.5	0.895151	0.895151
2	0.5	4.75	0.5	500	0.5	2.707651	2.707651
3	0.5	8	5.25	1000	10	13.520151	13.520151
4	5.25	1.5	0.5	1000	5.25	9.207651	9.207651
5	5.25	4.75	5.25	1000	0.5	23.488901	23.488901
6	5.25	1.5	10	1000	10	13.957651	13.957651
7	5.25	8	10	1500	10	46.770151	46.770151
8	10	4.75	5.25	1000	0.5	44.270151	44.270151
9	10	4.75	0.5	500	5.25	49.020151	49.020151
10	10	4.75	10	1500	10	53.770151	53.770151

Input Parameter	Minimum Value	Maximum Value
Hello Interval (seconds)	0.05	10.0
Hello Inactivity Interval (seconds)	1.05	8.0
ACK Timer Interval (seconds)	0.5	5.0
Maximum Transmission Unit (bytes)	500	1500
SPF Computation Interval (seconds)	0.5	5.0

Table 6: Full Factorial Model Input Plane.

generated from a single link failure. These paths we 11 and 12 hops long respectively, and the paths only varied at a single node. This means that in order to simulate the entire AT&T network for convergence times only required actually simulating 13 total routers. Obviously, this reduced the time required to run the simulation to the order of seconds. At this stage, the simulation requires a second or so to initialize, and on an average execution was complete in 0.0006 seconds. The simulation results we presented in this paper required 12,000 events to generate the convergence times in a simulation of 100 seconds.

Using this optimized model, we are now able to compare full-factorial experiment approach to RRS.

6 Analysis and Comparison of RRS to Full Factorial

In the previous section we showed how meta-simulation can reduce the amount of time to acquire meaningful results from our models by employing algorithms such as Recursive Random Search. Using RRS, we were able to generate all of our results in only 750 experiments, or simulation executions. However, RRS had not sampled a large area of the state space, so how confident can we be in the results?

We computed a full factorial model in order to validate the results we gained. We used the same 5 input parameters as in the RRS experiments, and 7 levels as shown in Table 6. In Table 7, we fitted a linear regression model to the data and found that the same three input parameters had the most effect on the model. The full factorial model produced an adjusted R-squared value of 85%. We were confident that RRS was properly modeling the same parameter space as we would have explored had we done a more detailed full factorial. Plotting the same two most effective parameters as we modeled in RRS, the scatter plot did not show any trends in the data, as shown in Figure 6. The final step in our verification was to create a quantile-quantile plot of the residual errors. We observed a linear relationship between sample error and theoretical. From Figure 6, the linear model assumptions of normality appear to be valid for the full factorial model.

This full factorial design generated 16,807 experiment runs, 20 times more than the RRS design, and yielded less information in the areas that we were interested in studying. Figure 7 illustrates the amount of detail generated by RRS versus the Full Factorial design for convergence times in the sub-second range. RRS also generate a ”good” value for the convergence time 0.11 seconds, which was within 7% of the Full Factorial design best value. While we could have generated a full factorial design using the final RRS input parameter ranges, we would not have had the benefit of the knowing that was in fact the area of interest, beyond our ability to analyze the system. In fact, we purposely chose the full factorial design presented here because we wanted to be certain about the nature of the system. It was necessary to explore a large range in order to validate our results

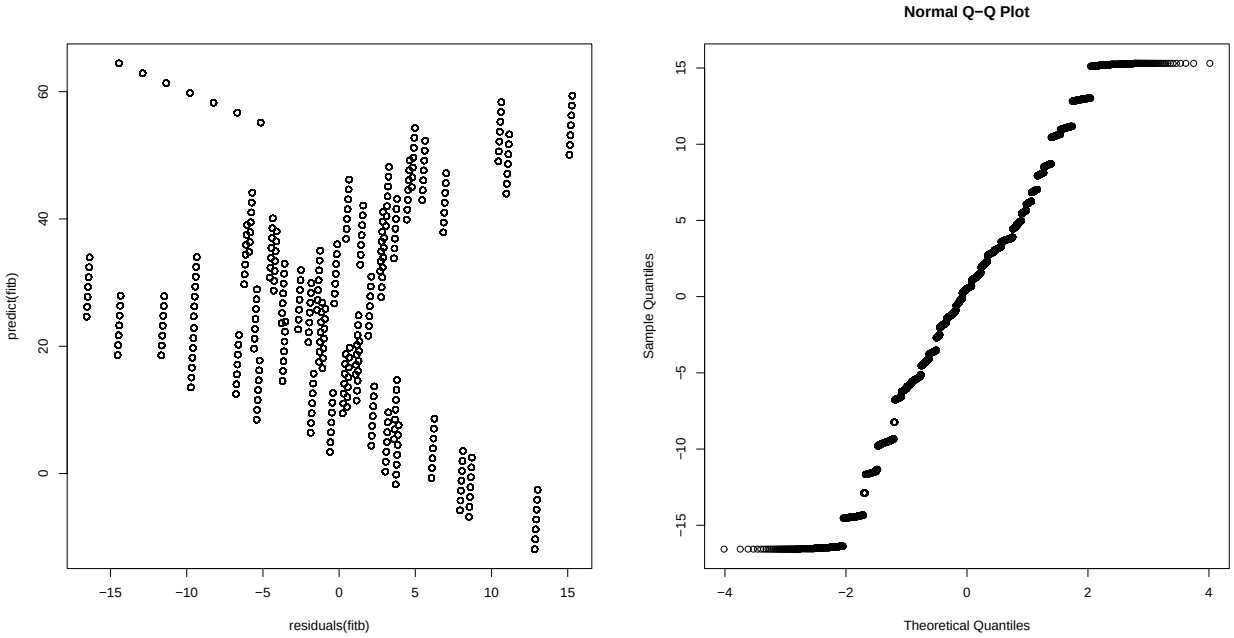

Figure 6: Full Factorial Design Scatter and Q-Q Plots.

Residuals

Min	1Q	Median	3Q	Max
-16.567	-4.262	0.523	3.762	15.306

Coefficients

	Estimate	Std. Error	t value	$Pr(>\|t\|)$
(Intercept)	-2.132e+01	2.472e-01	-86.26	$< 2e-16$
HELLO Packet Interval	3.844e+00	1.626e-02	236.49	$< 2e-16$
HELLO Inactivity Interval	4.690e+00	2.376e-02	197.41	$< 2e-16$
ACK Interval	-1.528e-17	1.626e-02	-9.40e-16	1
MTU	-1.335e-19	1.544e-04	-8.64e-16	1
SPF Computation Interval	9.796e-01	1.626e-02	60.26	$< 2e-16$

Residual standard error: 6.674 on 16801 degrees of freedom
Multiple R-Squared: 0.8543, Adjusted R-squared: 0.8543
F-statistic: 1.971e+04 on 5 and 16801 DF, p-value: $<$ 2.2e-16

Table 7: Full Factorial Model Output generated by R.

in the RRS design.

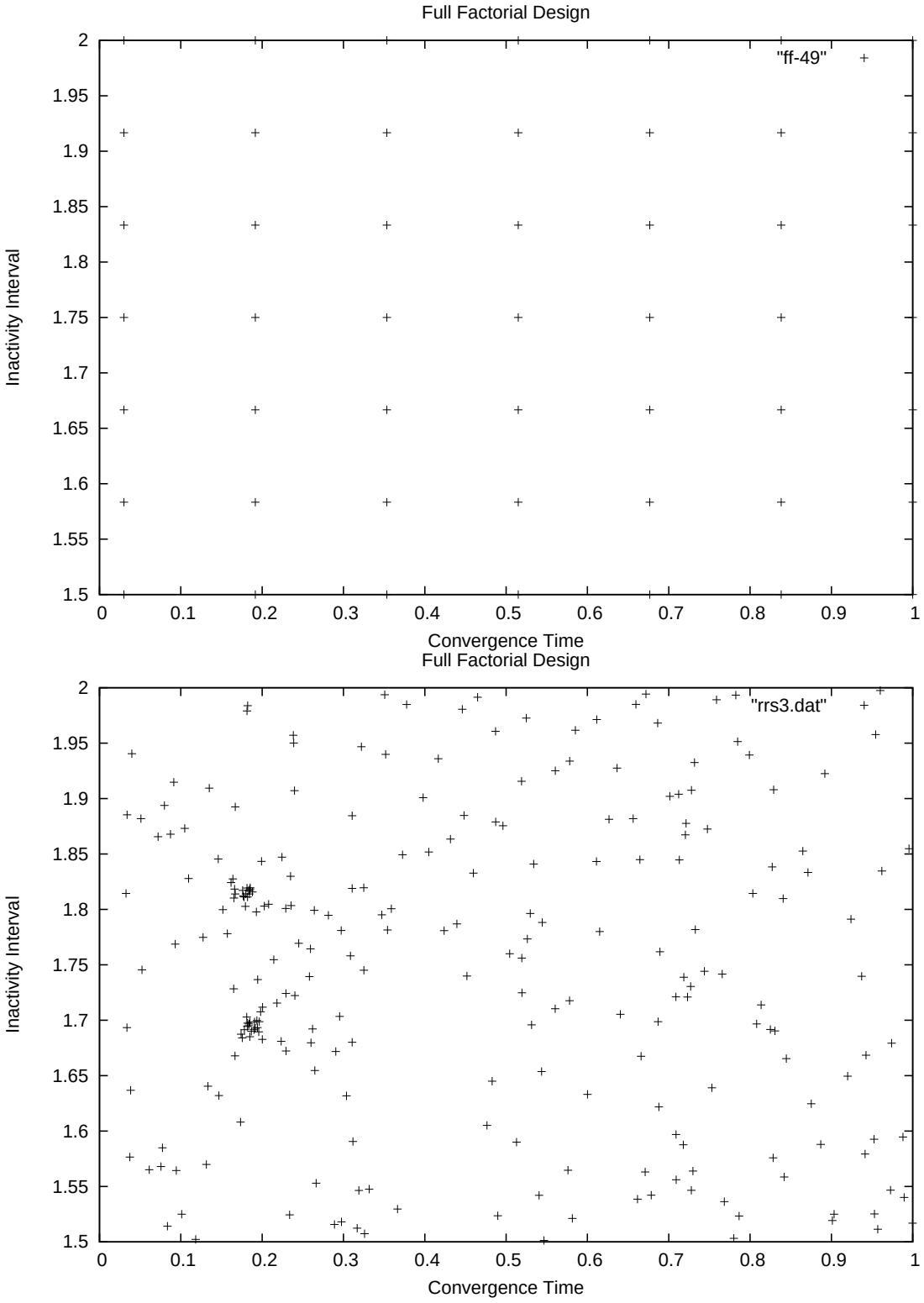

Figure 7: Full Factorial Design vs Random Recursive Search. The x-axis is convergence time (CT) and the y-axis is inactivity time interval.

7 Conclusions

In this paper, we demonstrate the efficacy of the Recursive Random Search (RRS) technique when applied to large-scale meta-simulation of OSPF routing networks. We found that:

1. the number of simulation experiments is reduced by an order of magnitude when compared to full-factorial design approach,

2. this approach enabled the rapid elimination of unnecessary parameters, and

3. RRS enabled the rapid understanding of key parameter interactions.

By using RRS we made the interesting observation that when modeling only OSPF control-plane dynamics we were able to shrink the number nodes down to that subset that was only needed for determining convergence times. This reduction resulted in models that execute 100 times faster than their full topology counterparts.

In the future, we plan to leverage our experience here to examine potential optimization to the OSPF protocol that may decrease convergence times over what has been previously reported.

References

[1] R. Jain, *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling*, Wiley - Interscience, 1991.

[2] S. Floyd and V. Paxson, "Difficulties in simulating the Internet," *IEEE/ACM Transactions on Networking*, February 2001.

[3] S. Floyd, "Simulation is crucial," *IEEE Spectrum*, January 2001.

[4] S. Floyd and E. Kohler, "Internet research needs better models," in *First Workshop on Hot Topics in Networks (HotNets-I)*, October 2002.

[5] D. C. Montgomery, *Design and Analysis of Experiments*, John Wiley and Sons, 2001.

[6] Z. Michalewicz and D.B. Fogel, *How to Solve It: Modern Heuristics*, Springer Verlag, December 1999.

[7] Aimo Törn and Antanas Žilinskas, *Global Optimization*, vol. 350 of *Lecture Notes in Computer Science*, Springer-Verlag, 1989.

[8] D. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, MA: Addison Wesley, 1989.

[9] E. Aarts and J. Korst, *Simulated Annealing and Boltzmann Machines*, John Wiley & Sons, 1989.

[10] T. Ye and S. Kalyanaraman, "A recursive random search algorithm for large-scale network parameter configuration," in *To appear in the Proceedings of ACM SIGMETRICS (part of FCRC)*, 2003.

[11] C. Alaettinoglu, V. Jacobson, and H. Yu, "Towards millisecond igp convergence," in *NANOG*, October 2000.

[12] W. C. Davidon, "Variable metric method for minimization," *SIAM Journal on Optimization*, vol. 1, pp. 1–17, 1991, The article was originally published as Argonne National Laboratory Research and Development Report May 1959(revised November 1959).

[13] L. Armijo, "Minimization of functions having lipschitz continuous first partial derivatives," *Pacific Journal of Mathematics*, vol. 16, pp. 1–3, 1966.

[14] P. Brachetti, M. De Felice Ciccoli, G. Di Pillo, and S. Lucidi, "A new version of the price's algorithm for global optimization," *Journal of Global Optimization*, vol. 10, pp. 165–184, 1997.

[15] R. MEAD and J. A. NELDER, "A simplex method for function minimization," *Computer Journal*, vol. 7, no. 4, pp. 308–313, 1965.

[16] R. Hooke and T. Jeeves, "Direct search solution of numerical and statistical problems," *Journal of the ACM*, vol. 8, no. 2, pp. 212–229, April 1961.

[17] C. D. Carothers, D. Bauer, and S. Pearce, "ROSS: Rensselaer's optimistic simulation system user's guide," Tech. Rep. 02-12, Department of Computer Science, Rensselaer Polytechnic Institute, http://www.cs.rpi.edu/tr/02-12.pdf, 2002.

[18] G. R. Yaun, D. Bauer, H.L. Bhutada, C. D. Carothers, M.Yuksel, and S. Kalyanaraman, "Large-scale network simulation techniques: Examples of tcp and ospf models," *ACM SIGCOMM Computer Communication Review (CCR)*, vol. 33, no. 3, 2003.

[19] Jiang Li and Shivkumar Kalyanaraman, "Ormcc : A simple and effective single-rate multicast congestion control scheme," in *Submitted*, http://www.cs.rpi.edu/˜ lij6/Research/index.html, 2002.

[20] G. Yaun, C. Carothers, and S. Kalyanaraman, "Large-scale tcp models using optimistic parallel simulation," in *In the Proceedings of the Workshop on Parallel and Distributed Simulation (PADS) (part of FCRC)*, 2003.

[21] T. Ye and S. Kalyanaraman, "A unified search framework for large-scale black-box optimiz ation," Tech. Rep., Rensselaer Polytechnic Institute, ECSE Department, Networks Lab, 2003.

[22] J. Moy, "OSPF version 2," *IETF RFC 2328*, April 1998.

[23] "Rocketfuel internet topology database," http://www.cs.washington.edu/research/networking/rocketfuel.

[24] J. Stewart III, *BGP-4 Inter-domain routing in the Internet*, Addison-Wesley, 1999.

[25] "The r project for statistical computing," http://www.r-project.org.

[26] O. Berry and D. R. Jefferson, "Critical path analysis of distributed simulation," in *Proc. 1985 SCS Multiconference on Distributed Simulation*, January 1985, pp. 57–60.

[27] M. Goyal, K. K. Ramakrishnan, and W. Feng, "Achieving faster failure detection in ospf networks," in *Proceedings of the International Conference on Communications (ICC)*, 2003.

[28] T. Ye and S. Kalyanaraman, "A Recursive Random Search Algorithm for Large-Scale Network Parameter Configuration", in *Proceedings of SIGMETRICS*, pages 196–205, 2003.

ACKNOWLEDGMENTS

This research is supported by an NSF CAREER Award CCR-0133488, the DARPA Network Modeling and Simulation program, contract #F30602-00-2-0537 and a AT&T University Relations Program Grant.