Engineering Optimization Models at Runtime for Dynamically Adaptive Systems

Fahad Javed and Naveed Arshad
Department of Computer Science
LUMS School of Science and Engineering
Lahore, Pakistan
fahadjaved (at) lums.edu.pk
http://na.lums.edu.pk

Fredrik Wallin and Iana Vassileva and Erik Dahlquist
Department of Public Technology
Mälardalen University
Västerås, Sweden
fredrik.wallin@mdh.se
iana.vassileva@mdh.se
erik.dahlquist@mdh.se

***Abstract*—Dynamically adaptive systems (DAS) such as smart grids, cloud computing applications, sensor networks and P2P networks tend to change their structure at runtime. Therefore, design-time modeling for such systems are sometimes not enough to incorporate self-* properties. To this end, we have developed a dynamic mathematical modeling framework for runtime optimizations for DAS. In this paper, we describe how our system engineers a linear programming model by using a smart-grid application for power distribution as a case-study. At runtime whenever an optimization is desired this modeling framework captures the state of the system, converts it into an appropriate linear programming model, plan the changes using mathematical manipulations and apply the changes to the actual system. Our results show that this framework is able to capture accurate runtime models of large power systems and is able to adapt itself with the change in the size or structure of the system.**

I. Introduction

Dynamically adaptive systems (DAS) or autonomic systems such as smart grids, sensor networks, cloud computing systems pose new research challenges to the modeling community. These challenges often stem from the fact that usually DAS have to adapt to changes not envisioned at design-time. Often these changes are to be carried out under strict time-lines. Since these are time critical changes, often they have to be carried out automatically by the system itself without any human involvement. Indeed a system cannot know about itself unless it has the property of self-awareness. Just like in humans pulse-rate, rate of breathing, temperature, etc. provide an awareness of an impending illness, similarly a system requires self-awareness to be able to detect any impending problems in the system.

Since DAS change their structure and size frequently, a fixed design-time model may not be enough to achieve self-awareness. Moreover, even if the system does not change itself the environment in which the system operates may require the system to change its behavior. In both cases the system has to adapt itself. This adaptability is not possible unless the system not only is self-aware but is also aware of its environment.

To this end, self-aware and self-composing modeling frameworks for runtime modeling are needed. In this paper we have used a mathematical modeling technique i.e. linear programming to develop a runtime model of the system to achieve this goal. This paper uses model-driven engineering based on runtime modeling for self-optimization. Specifically, our modeling framework makes the following contributions:

First, our framework is based on industrial strength operations research technique i.e. linear programming that has the capacity to handle hundreds and thousands of decision variables, thus it is an ideal candidate for modeling DAS. But traditionally linear programming models are constructed at design time. Thus, as our second contribution, we have developed a runtime model composer for linear programming that generates a runtime model of the system and its environment instantaneously. Our third contribution is that the runtime model is used to carry out optimizations in the system automatically using mathematical manipulations, thus making the system self-optimizing.

II. Motivation: Power Conservation in Smart Grid

In this section, we introduce a motivating example that requires dynamic model generation for self-optimization. We also use this running example during the rest of the paper to illustrate our approach.

Power conservation has been an important issue for many decades now. The scarcity of the energy sources and the investment needed to setup new power sources has been pushing up the cost of power. Even today power rationing is being enforced in countries where power demand is growing much faster than the growth in the production capacity. In this scenario the research community is working hard to find the solutions to find new inexpensive and renewable sources of power. Additionally, another research effort is to conserve the existing sources of power as much as possible.

Within the scope of power conservation, our previous work has focused on reducing the demand for power while maintaining a service-level guarantee for the subscribers [14]. On one hand this means adequate power availability to the consumers and on the other hand this means lower power prices for power generation companies.

Specifically, we have focused on managing power to devices of higher consumption. Our hypothesis is that if we can micro-manage power to the most power consuming devices

such as heating and cooling units then we are able to save the maximum energy. To satisfy the needs of the users, in this approach there are service-level guarantee for power availability . This service-level guarantee allows the user to use the power at specific time schedules that is provided by the power company.

In such a system where there are hundreds of thousands of high powered devices, without a runtime model of the system managing the service-level guarantee is a huge problem . Moreover, the number of devices that are available at a certain time may fluctuate. Therefore, modeling such a system at design time is almost impossible.

In this paper, we propose a runtime modeling framework that provides us with an instantaneous runtime model of the system. This instantaneous model is created by converting the raw-data from the system to a mathematical model, i.e. linear programming model, of the system. Through this mathematical model that is generated at runtime, we are able to conserve power as well as fulfill all the service-level guarantee for the users.

At this time it is appropriate to discuss other research efforts related to this work.

III. Related Work

The related work can be characterized into three separate categories.

First, is the related work in the field of modeling runtime systems. There are various modeling approaches for instantaneous representation of a system. Various runtime models are proposed for various purposes. For instance, the focus of modeling in the runtime modeling community has been on capturing architectural or operational modeling of a system and representing it in a common format such as UML or XML. In this regard, Kuhn and Verwaest developed a polyglot library for modeling a computing system at runtime [16]. This runtime model generates and annotates a UML diagram for architectural view at runtime. Combemale and colleagues have designed a modeling framework using a markup language for autonomic transformations of system [3]. Their system uses an XML derived language to implement model based-adaptation. Sanchez and colleagues and Cetina and colleagues have used extensions of UML for similar purpose [2],[23].

In general, these modeling at runtime techniques are geared towards extracting a model from an existing system in UML or XML formant and then transforming the system based on Model Driven Engineering (MDE). The goal of our research is similar in that we need a runtime model of the system. However, we would also like to manipulate the model to achieve self-optimization. Although our current work focuses on extracting a model and representing it in mathematical form, One possible future direction of our work could be to transform UML or XML format into a mathematical model.

In this context an interesting work was proposed by Dobson and colleagues [4]. In this work the authors explored the idea of modeling the system as a whole. In our system we follow a similar paradigm where we model the observed nd the controlling parameters in a mathematical model to achieve the desired adaptation.

The second category of related work is the existing work on optimization of self-managing systems. This work can further be divided into two broad categories: Top-down modeling and bottom-up modeling: In top-down modeling, a global model of the system is composed by modeling the individual artifacts of the system. For example, Zhu and colleagues and Gounaris and colleagues optimized systems on the global model of the system where the model artifacts were aggregations of individual components such as demand and supply [26], [8]. Optimization have used linear programming, such as our previous work and works of Femal and Freeh [12], [5], control theory by Lefurgy and colleagues and Wang and colleges [17], [24] and game theory by Khargharia and colleagues [15] for their global level optimization. However, all these methods are for fixed sized systems and do not provide the ability for the model to grow or shrink at runtime.

Bottom-up techniques, in this context are more suitable. Optimization for wireless sensor networks using learning has been proposed by Shah and Kumar[22] as well as by Mainland and colleagues [18]. Lefurgy and colleagues and Nathuji and colleagues have optimized individual server power load using control theory [17], [19]. However, not all problems have the property of sub-problem optimality. For problems which involve planning over a period of time, a holistic view of the entire system is needed to ascertain the optimal solution. In essence any problem for which greedy algorithm is not suitable, for similar reasons bottom-up technique will not be optimal either.

The third category of related work are the approaches to conserve power in large scale systems. Optimizing computing system for power conservation has been done since long. However, we would like to state here that our work is NOT optimization of power systems in the traditional way. In traditional power supply optimization the engineering aspect of power and its distribution is taken into account. But in our work we are managing the power devices that make up the smart grid.

The closest work in optimization of power using end user devices is optimization of power dispatches and similar issues by Wang and colleagues [25]. This work provides a method to optimize power dispatches. However, the method is still far from optimizing device usage. The optimization uses a fixed model of system as in this specific problem the number of variables in the target system do not vary.

Few researchers have successfully optimized large scale power systems. However, their assumptions for their target systems do not hold for power distribution grids. Femal and Freeh [5] optimized power using linear programming for a data-center. However, for their model number of devices had to be known at design time.

A more related work is by Hlavacs and colleagues [9]. However, their aim of optimization was to manage the optimizing agent's power consumption rather than optimizing the system itself.

Another interesting dimension of runtime modeling of DAS is work of Goldsby and Cheng [7]. However, the modeling is focused on handling uncertainty and will not be of direct use in optimization.

Thus though optimization is a very active field and research has been done on optimizing a modeled system, but this is one of the first attempts to model an evolving system, such as a power grid.

IV. Modeling Framework

Traditionally, models created for optimization of systems are generally expressed as abstract mathematical models. These models are defined in standard mathematical lexicon. When a system is to be deployed, its model is realized as code segments or equation matrices or equation arrays based on solver being used for optimization. The dimensions of these matrices and cardinality of variables is usually defined at the time of deployment and is hard coded in code segments, matrix dimensions, etc.

In comparison, for systems such as DAS, system dimensions at the time of deployment are meaningless . This is due to the fact that it can grow, as well as shrink over time. To handle such changes, a measure of self-aware modeling integrated with self-optimization is necessary to manage DAS. This self-aware optimization can leverage the change in dimensions of DAS at runtime to attain scalability and performance boost according to the runtime state of DAS.

Various systems have been optimized through mathematical models. However, in all of the applications of mathematical techniques seen so far by the authors, the constraints and tuning parameters were known when the system was being implemented [5], [12], [11]. We have not observed any detailed work for engineering a system's model that exhibited variability in the size of their constraints and control features.

Therefore in our modeling framework we have used the abstract mathematical models as a meta-model to create an on-demand, instantaneous model of system based on system statistics. In this section we define our modeling framework for constructing an instantaneous model of a system at runtime.

A. Structure of the Mathematical Meta-Model

In practice mathematical models are developed and expressed as abstract models. Mathematical models represent a system in form of decision variables and constraints. Decision variables are the controlling parameters to change the system state where as constraints are the limitations of the system. Since in mathematics, a variable can take any numeric value, it is important that we specify the limits of our decision variables as well.

To model a system, the control parameters and limitations of the system are analyzed. A system can be composed of many control parameters but usually there exist logical groupings with which these control parameters can be abstracted into a single entity or class. Usually this also means that similar constraints apply on each of the element of the grouping. It also means that a single abstract equation with appropriate quantifiers can suffice for containing the behavior of all the variables within a group. Since these are logical groupings and resemble a set like structure, we call these variable abstraction as ontologies of our system. Hence an ontology is a group of control parameters which have similar logical structure and are subjected to similar constraints. Like sets, ontologies can be grouped together to form more inclusive notation. Mathematically, this means that whereas two different logical groups of variables, or ontologies are subjected to their own constraints, it can also have a set of constraints that are applicable to both the groups. Hence our decision variables can be part of a multitude of ontologies. Here a subscript define the specific element within an ontology. We call these grouping of ontologies as an ontological class. Figure 1 describes the abstract model that we will discuss in detail here.

Example: Model for a Smart Grid Application
We take the example of modeling the usage of electrical devices in an electric grid. We divide our devices into ontologies according to their consumption profiles and time periods. Our task is to maximize the number of machines from each set which can be kept in "on" state for a particular period in an hour without violating the service-level guarantee. Here the number of machines to keep in "on" state in a particular time period is our tuning parameter or "decision variable". For each tuning parameter there are two ontologies. First there are different sets of machines. Each type is represented as a subscript i. The second attribute is of time, that is which time period does a specific decision variable represent. These types are represented as a subscript t. Hence i, and t represent two ontologies combined in a single decision variable $X_{i,t}$.

The system in figure 1 is subjected to three classes of constraints. Each of these class is represented as a single abstract equation. Notice that equation 3 is only applicable to one ontology, the time t while the other two are subjected to both. For demonstration of our framework we will consider the example of equation 2 in detail. This equation constraints the system by enforcing a minimum service level. It states that for every time period t, the number of machines switched on in every machine class i should not be less than $1/3^{rd}$ of the total number of machines in that class.

During implementation these abstract models are expanded according to available system statistics. If our system had fixed machine classes, say 10 and 6 time periods (t) the abstract equation 2 would have been expanded to 60 equations. Each of these 60 equations would have represented one specific (t, i) tuple.

Mathematical models for systems which do not exhibit change in cardinality from abstract model to implemented model can be modeled effectively. That is, if we can enumerate at time of implementation or deployment as to how many machines we have and how many time segments we have, then generating an actual model of the system from abstract model is straight forward.

However, if the cardinality cannot be evaluated at the time of implementation, then modeling becomes a difficult task. A naive modeling technique is to consider worst case scenario.

For example, in the sample model above, we limit i, or device classes, to say 1000 and then make a model for these many classes.

For a grid level electric distribution network this solution is not feasible. First, the number of device classes cannot be predicted. There are new types of machines that are being added everyday and limiting this growth is not possible. Second, worst case setup is highly inefficient. By calculating for a 1000 classes always, we are consuming much more resources where as in actuality we might need a fraction of these calculations. Third, because we always assume a large data-set, the choices for algorithms is limited. There are algorithms which are more efficient for small to medium sized data-sets. If we can evaluate and model at runtime, it is possible to derive a better result by using more accurate algorithms.

B. Modeling at Runtime

Various techniques exist for creating a runtime model of a system. These efforts are usually intended for architectural and operational runtime modeling systems. We observed that these modeling framework have some commonality in processing their task. Usually runtime modeling frameworks define a set of primitive artifacts with defined semantics. At runtime these artifacts are instantiated and replicated and relationship among these artifacts is established [21], [7], [16]. There are various methods to extract information from a system and various uses of the modeled systems, but this is beyond the scope of runtime model generation.

The underlying architecture of our framework is similar to these runtime modelers. The difference is that we use the components of abstract mathematical models as our primitive artifacts. Specifically, the abstract mathematical model defined for the system is used as a meta-model. The primitive artifacts for us are the ontological classes. When we observe an object, or a variable, belonging to a specific ontology, we create a corresponding ontology object for it in our mathematical model. This process is covered in the modeling of ontologies step (step 1 defined below).

The equations of our meta-model define the relationships between different variables. Once we determine the cardinality of ontological classes, we develop relationships of ontologies by exploring the equations one by one and setting up the constraints and limitation of the system in the process. This process and production of the complete model is generated in the modeling phase.

$$Maximize(Z = \sum_{i,t} X_{i,t}) \tag{1}$$

$$\forall_t \forall_i X_{i,t} \geq supply_i/3 \tag{2}$$

$$\forall_t \sum_{i,t} \mu_i X_{i,t} \leq supply_i \tag{3}$$

$$\forall_{i,t} X_{i,t} \leq MAX_i \tag{4}$$

Fig. 1. Hourly planning LP equations

This runtime modeling is three step process. Our framework first determines the system statistics to define cardinality for ontologies. In the second step, it determines the cardinality of relationships and determine the number of equations each meta-equation will generate. The third step uses the cardinalities to create an instantaneous model. The second and third steps are closely related and their implementation is also intertwined. However, since step 2 is platform independent and step 3 is dependent on the solvers, merging the two steps is avoided where-ever possible.

The description of the phases is given below.

1) Modeling of Ontologies: Modeling of ontologies is a two step process. First we pre-process our data to reduce dimensions of our input data.

An input to our system consists of raw usage data for devices. In pre-processing we reduce the dimensionality of raw usage data using a clustering algorithm. The details of this dimension reduction is discussed in our previous work [14]. This pre-processing is required due to the nature of problem. In other works such as Femal and Freeh's use of LP, such pre-processing will not be required[5]. For such models, direct evaluation is possible.

Modeling of ontologies determines the cardinality of each ontological class. In our model, there is only one ontological class, X. This ontology in turn is composed of two co-dependent ontologies: time interval, represented by subscript t and instance of a cluster represented by subscript i. We consider 6 time intervals for our problem, however, this interval can also be changed in runtime.

2) Modeling of Relationships: A mathematical model is a representation of system in terms of inequality and equality equations. These equations define the constraints and limits of the system.

Our framework first distinguish between the equality and inequality equation. Though both are evaluated in the same way but in construction step, a different matrix is generated for each of those equation genres.

Our framework in this step uses cardinalities of ontological classes to expand the quantifiers. Each quantifier expands some ontological classification. For example, a $\forall X_i$ quantifier translates to 1 equation for ontology i within the ontological class X. In addition, the co-efficient and right hand side for these equations is also determined in this step as constants are sometimes also associated with a specific instance of ontology.

Similarly, equation 2 has a $(\forall_t \forall_i)$ quantifier. Hence this meta-equation is expanded into i x t equations, since the equation is created for each (i, t) tuple. The equation states that the coefficient of $(i, t)^{th}$ decision variable is 1. So for each new equation expanded for meta-equation 2, the coefficient for variable $X_{i,t}$ will be one and all other variables will have coefficients of zero. The equation states that the right hand side of this equation will have the constant value of $supply_i/3$. The $supply_i/3$ is the cluster of the set X_i. We determined this value in step one. Hence for each equation the correct

corresponding value for $X_i/3$ is placed.

3) Model Construction: A mathematical model can be represented in different forms. One of the most commonly used form to represent mathematical models in computing systems is a matrix form. Since arrays and matrix are realization of the same phenomenon, we will discuss how we created matrices from our results from previous steps.

In matrix notation, a series of linear inequality equations are represented as:

$$A \times x < b$$

and a series of linear equality constraints as:

$$Ae \times x = be$$

Here x is a vector representing the variables, b is a vector for right hand side constants for inequality constraints and be for right hand side constants for equality constraints. Similarly A is matrix of coefficients of x for inequality constraints and Ae for equality constraints. Similar generalizations exist for non-linear systems but is beyond the scope of this work.

Though both equality and inequality constructs are almost similar but solvers accepts them in two different set of matrices. We construct both the matrices in similar fashion.

The process of constructing matrices is as follows: We first determine the matrix x. We use the notion determine because x is not constructed in matrix form per se. Rather x is considered as an ordering of decision variables. Decision variables, if we recall, are the instances of various ontological classes that we created in step 1. Fixing the order does not change the execution of algorithm so any convention which completely covers the ontological class space is sufficient. However, fixing an order is necessary as this order determines the placement of coefficients in matrices A and Ae.

Our model has a single ontological class of decision variables, $X_{i,t}$. We fix an order of expanding the two dimensional space of X by arranging rows before columns. This step fixes our x vector.

Our framework proceeds with processing our equations determined in step 2. For each equation a row in matrix A and one in b is added for an inequality constraint. Similar step is executed for equality constraint but for matrices Ae and be. In a newly added row of A, all elements are zero excepts the ones specified by the equation. The constant values for coefficient of A and the value in b are placed. This step is repeated for all the equations which were generated in step 2. At the end of this The complete matrices A, Ae, b and be are produced.

C. Running Example

We now describe the construction for a row of equation 2. Let's assume that we 50 clusters were created during our preprocessing and we have 6 time slots. This means that our step 1 will provide us with the value of 300. These are the number of decision variables that we will have in our system. For our model generating step, this means that size of x matrix will be 1×300 and matrices A and Ae will have 300 columns.

Lets assume that cluster number 10 has 18 elements. So our equation for second time period from step 2 will look like the following:

$$1 \times X_{10,2} < 6$$

Our model construction will construct the following row for this equation in matrix A.

Column	1..61	col 62	63 .. 300
Value	0..0	1	0..0

In addition, it will add a row in matrix b and put the value 6 in the newly added row.

A complete matrix A thus will have $x \times t$ columns and $i \times t$ equations for meta-equation 2, t equations for meta-equation 3 and $i \times t$ equations for meta-equation 4 and a solitary equation for meta-equation 1.

V. Evaluation

We have designed a framework for modeling of optimization of large scale power systems. Conservation of power through optimizing usage of end user devices is a not new concept. However, to our knowledge very few techniques are available which are scalable and efficient to achieve this goal. So far the major work in this field has been performed on fixed sized systems where the number of devices are known at design time. The models of systems are before deployment time based on the largest possible or worst case deployment of system [1], [6].

Our system engineers the model at runtime instead of populating the variables of a fixed model. Therefore our evaluation, compares the existing modeling methods for similar smart-grid application with our runtime modeling results. We claim improved performance using two key matrices; First our response time is faster than a fixed model. Second, we claim better efficiency in achieving goal of optimization, i.e. in distributing power to the consumers.

The aforementioned 'efficiency' of our electric distribution is the unutilized power (UP) in the system that an optimization is unable to distribute amongst the electric devices. The details of why such unutilized power exists is discussed in our previous work [14]. We would like to state here that the increased efficiency in our example system is because we modeled it in a way so that a decrease in model size will increase efficiency. Thus our results of efficiency are applicable when the system can be and is modeled in a way which relates the efficiency with size of model.

Our evaluation thus evaluates the following hypothesis: Does modeling at runtime for a system that varies its size and structure results in benefits in terms of time or efficiency. In order to test this hypothesis we used two sets of real data collected from two different sources.

We conduct our evaluation on two different sets of actual data readings. This is because of two reasons: First, consumption data of individual users for a city is not readily

available. Second, this split analysis proves applicability of our framework for systems both large and small.

Our first set is a small but detailed study of household energy use in Sollentuna, Sweden performed over the course of two years. Experiments on this data is used to show a correlation between total consumption, time to calculate and the number of users. Our second experiment is on a data from the state of California, USA. In this experiment we apply our modeling framework on large scale set of data and we see the benefits in terms of efficiency.

A. Evaluation Setup

For our evaluations we used a shared 2.4 G.Hz. Pentium Core 2 Duo processor with total of 2.00 GB of RAM. The mathematical solvers used Matlab's optimization toolbox.

B. Evaluation Data Details

Our first experiment uses hourly consumption data from approximately 700 houses collected in Sollentuna, Sweden for the year 2005-2006. Through this experiment we validated the following

- There exists a strong correlation between the time taken for optimization by dynamic modeler and the consumption of energy.
- There is a weak correlation between a fixed model optimization and consumption of energy.
- there exists a strong correlation between total demand for energy and the number of consumer clusters.

Whereas the first two claims support the case for dynamic modeling, the last claim helps us construct a more powerful scenario for validating the scalability and applicability of our modeling framework.

Our modeling framework can model and optimize systems which vary in size. The real benefit of the system is attained when the variation in size is considerable and the scale of optimization is large. Since a small scale LP optimization in itself takes insignificant time. To test our framework for a large scale realistic system we use data published by CAISO. This data consists of daily usage of electricity in state of California, USA. A sample of this data is provided in figure 2. However, this data is incomplete for our modeling since we require the usage pattern of individual users and not just the total consumption of the system. To overcome this problem, we artificially constructed the clusters of users based on total consumption by dividing the total consumption over in a Gaussian distribution. Gaussian distribution was used because it was the most appropriate and simple distribution to represent the natural behavior of large number of users. Though the distribution of load has a minor impact on the overall performance. We still consider it as part of our future work to model and evaluate the system with different distributions. To validate this distribution further, we used results from our first experiment set. Even though, intuitively it makes sense that increased consumption means increase in number of consumers. We still base our argument for constructing the usage patterns for individual users based on the correlation found between consumption and users in our first experiment.

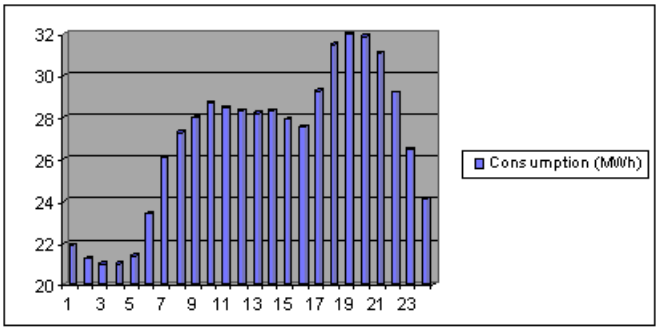

Fig. 2. Consumption profile of California for a day as published by CAISO (Consumption in MWh)

In the following sections we define the standard modeler, the modeler simulating the prevalent modeling methods in smart-grid literature, and our dynamic modeler using the aforementioned sets of data. The first set of data will validate the correlations and the second set will validate the scalability and efficiency of our framework in a large scale environment.

C. Standard Modeler

Smart-grid techniques which focus on global optimizations such as in [1], [10], [11], and [25] build models for the worst case scenario. Without a runtime modeling framework this is necessary because updating the system model manually at runtime is not possible.

For a system such as our micro-management application for smart-grids, a model using the standard method means constructing a model for the worst possible day throughout the life cycle of the system. Instead of simulating this scenario, we only consider the cluster configuration for the worst hour of the day we conducted our experiments on. Note that this is not the worst case or largest configuration for the system life cycle. However, this provides sufficient comparison since our technique has proven itself to be faster. We use the number of clusters as the metrics here because the size of the model is dependent upon the number of clusters for each hour. We used standard k-means clustering on the input data where k is worst-case clustering size for the day. These k clusters and their frequencies populate the fixed input matrix for the optimizer.

D. Evaluation Results

1) Swedish Household Consumption Data: Our first experiment uses the data collected from Sollentuna, Sweden. The data consists of consumption of electricity in a suburb of Sollentuna collected at an interval of 1 hour. We use these consumption profiles as input for both our dynamic modeling framework and the standard modeler. We conducted the experiments multiple times and considered the mean of runs to deal with operating system related noise in response time. This is because the response time for the small data-set is small enough to be affected by background processes of the operating system.

The execution time for the dynamic modeler, standard modeler, and the total demand of the system is shown in figure 3. Here the line with square points represent the time for standard modeler in seconds, the line with diamond points represent the response time of dynamic modeler and and the line with square points represent the total energy demand in MWh. As it can be observed, there is a correlation between the demand and the response time for the dynamic modeler. The correlation coefficient for these observations is 0.75 using Pearson method. On the other hand, the relation between response time of standard modeler and demand comes out as week inverse, -0.3 using Pearson method. This validates our first two claim that the strong correlation exists between time taken by dynamic solver and the total demand of the system and that a fixed size modeler is not able to benefit from the change in demand.

Our third claim is explained through the graph in figure 4. Here the dotted line represents the total demand for each hour, the line with square points represents the number of users and the strong line with triangle points represents the cluster count. Here we can see the relation between the number of consumption clusters and the total consumption. We see that a strong correlation exists between the number of and the total consumption using Pearson method (0.83).

We can thus conclude from this experiment that a strong correlation exists between the consumption, number of users and time taken by the dynamic modeler. Furthermore, no correlation was observed between the standard modeler and total consumption.

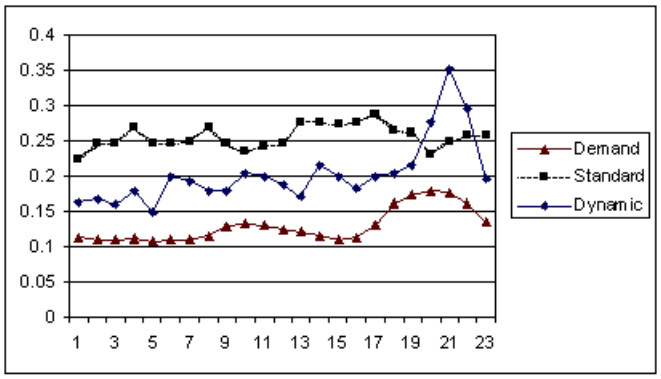

Fig. 3. Response time for dynamic and standard modeler in comparison to demand. (Response time in seconds)

2) CAISO Data: Our second evaluation compares our modeling framework with the standard modeling method on the criterion of running time and efficiency if we were to distribute electricity in state of California using our method. We evaluated our system by running both systems on data of 24 hours from a power distributor's profile.

We observed that our framework's execution time was considerably less in comparison to the standard modeler. Figure 5 plots our framework time and standard modeler time. Here the squares represent the time in seconds the standard modeler required to model and optimize the data for that specific time

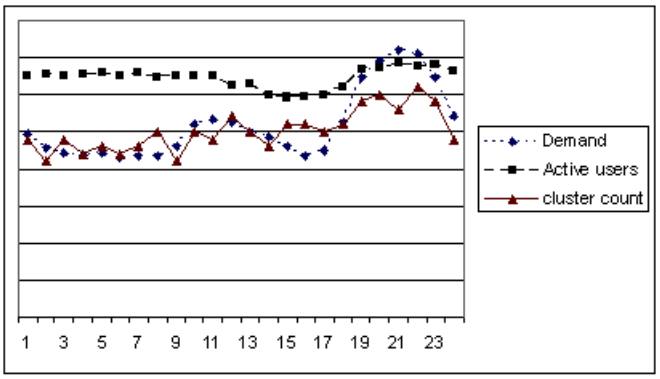

Fig. 4. Comparison between demand, clusters and active users for 24 hour period as observed in Sollentuna, Sweden

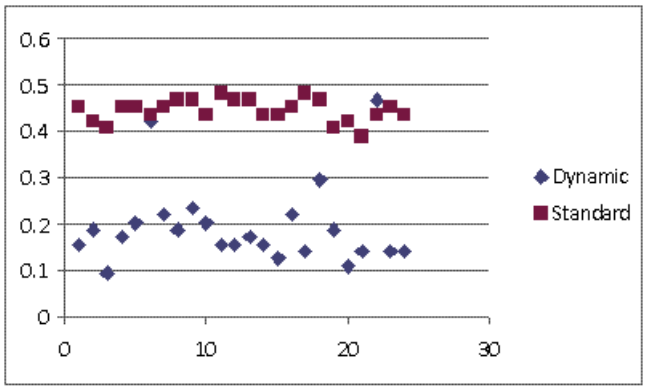

Fig. 5. Solver time for 24 hours CAISO data. (Response time in seconds)

period and diamonds represent the time in seconds for our runtime framework. It can be observed that runtime modeling time is considerably faster throughout, except for two cases, in the 6^{th} and the 21^{st} periods. These are the cases where the size of runtime model was maximum and both the models were of similar size. We witnessed at an average 56% better response time than the standard system.

Our second evaluation goal was to achieve better performance. Figure 6 plots the power allocated by runtime framework and by standard modeler. Here diamonds represent the runtime framework allocation of power in megawatts and squares represent the standard modeler results . We observed a marginal improvement in allocation of power. Total increase in power allocation was close to 2% which is significant for a large scale system.

Our results show that our runtime modeling framework is faster than a static modeling method. Our runtime modeler is approximately 50% faster than the standard modeler. Furthermore, we observed that we can achieve better performance for our specific model through the use of dynamic runtime modeling

VI. Future Dimensions of Runtime Modeling

Dynamic modeling intuitively leads to an efficient of optimization. Since the model only consists of variables and

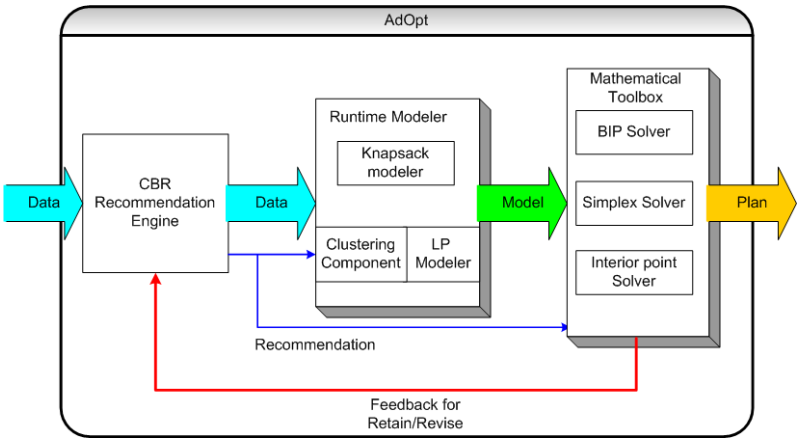

Fig. 7. System Flow [13]

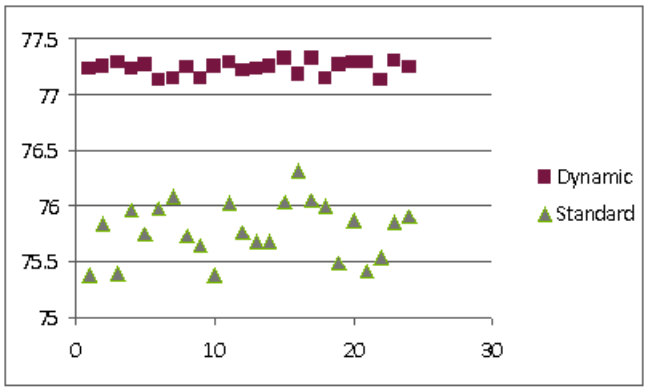

Fig. 6. Solver efficiency for 24 hours CAISO data. (Power allocation in Kilo-Watts)

constraints that are applicable at that instance, a more streamlined and concise model is constructed resulting in faster optimization and better results.

However, engineering a model dynamically is a powerful tool when seen in the context of adaptable optimizations and this is worth further discussion. Our dynamic modeling framework, is an extension of our previous work known as AdOpt technique [13]. In AdOpt, as shown in figure 7, based on the input parameters we select one of the many optimizers available, create a model for that optimizer at runtime and build a plan for execution. AdOpt uses a fixed set of ontologies which are more or less pre-defined, with homogenous consumption patterns.

From the study of application of smart grid, cloud computing and other fields where adaptable behavior is anticipated in future systems, we see that this rigidity of structure will not be guaranteed in our future systems. We have seen optimization applications of smart-grid such as applications for Plug-In Hybrid Electric Vehicles (PHEVs) [6] where the demand pattern of the users is an evolving phenomenon. From modeling perspective this means that the relationships and constraints for the system will be described at runtime. Even more appropriate comparison is the work of Ogston and colleagues who define an adaptive clustering method to group together various devices [20]. The technique is scalable for clustering devices in a city and the resulting clusters, their patterns, frequencies and shape will emerge at runtime. If we are to use this data to manage these devices then runtime engineering of model that considering the new clusters and patterns will be necessary.

Our modeling framework provides the basis for engineering models for such techniques of the future. Although our existing work caters for LP. However, the three step engineering process described in section IV-B for creating models is more or less same for modeling non-linear, integer and some heuristic optimizations. Our work not only provide solution for the smart grid problem but also provides a foundation for future dynamic modeling for these modeling techniques.

Our current framework is a proof of concept and requires engineering to integrate a meta-model into our framework. In our future work we look at ways to bridge this gap. We are working on evolving a method to define mathematical abstract models in a language which our framework can understand and create a meta-model from. To this end we are evaluating various modeling languages and are planning on including a translation engine which will translate abstract mathematical equations into a meta-model. Such work will streamline integration of our framework with existing optimization platforms.

Our second direction is looking at ways of determining constraints from system statistics. In our current framework, system cardinality of system constraints are determined solely by the cardinality of quantifiers. However, systems which can "sense" constraints through statistical analysis can produce much more powerful modelers.

Our third direction of interest is integration of our framework and optimizers with physical infrastructure and implement optimization of resources. A running system of this sorts will be of real benefit to society.

VII. Conclusion

Modeling methods for planning conservation for large scale systems are very few. Whereas the dimensions of a power supply grid change frequently, optimization models that can adapt accordingly are fewer. In this work we have proposed a framework which can self-adapt to the scale of the problem. This self-adaptation is done by expanding meta-models at runtime to develop an instantaneous model of the system. The expanded model is represented in a matrix form for use for various optimization toolboxes which are available in the market.

The advantages of our method is the increase in scalability and efficiency and the low learning curve to adapt our model into an existing optimizing engine. We base these claims on following reasons. First, because we make the model at runtime, we can scale up as the situation demand. We can also scale down and increase our efficiency. Second, we use the abstract model made to define the system for our framework. Since a basic model is a need for any optimization, there is no additional training or learning required to make any model adapt to our framework.

In conclusion, we have defined a framework to develop an instantaneous model of a system at runtime. This system can be used by large scale systems which change their structure at runtime. Example of these systems can be managing of end users devices within an electric power grid. The framework adapts the optimization model according to statistics of the system. Our evaluation results show that this adaptation aids in optimization and reduces the time of optimization by 56%.

VIII. Acknowledgement

This work is in part supported by grants from the Department of Computer Science at LUMS, ICT Research and Development Fund of Pakistan, and Higher Education Commission of Pakistan.

References

[1] S. Ashok. Optimised model for community-based hybrid energy system. *Renewable Energy*, 32(7):1155 – 1164, 2007.

[2] C. Cetina, P. Giner, J. Fons, and V. Pelechano. A model-driven approach for developing self-adaptive pervasive system. In *Workshop on Models@Runtime 2008*, 2008.

[3] B. Combemale, L. Broto, X. Crégut, M. Daydé, and D. Hagimont. Autonomic management policy specification: From uml to dsml. In *MoDELS '08: Proceedings of the 11th international conference on Model Driven Engineering Languages and Systems*, pages 584–599, Berlin, Heidelberg, 2008. Springer-Verlag.

[4] S. Dobson, E. Bailey, S. Knox, R. Shannon, and A. Quigley. A first approach to the closed-form specification and analysis of an autonomic control system. In *Engineering Complex Computer Systems, 2007. 12th IEEE International Conference on*, pages 229–237, July 2007.

[5] M. Femal and V. Freeh. Boosting data center performance through non-uniform power allocation. *Autonomic Computing, 2005. ICAC 2005. Proceedings. Second International Conference on*, pages 250–261, 13-16 June 2005.

[6] M. Galus and G. Andersson. Demand management of grid connected plug-in hybrid electric vehicles (phev). In *Energy 2030 Conference, 2008. ENERGY 2008. IEEE*, pages 1–8, Nov. 2008.

[7] H. J. Goldsby and B. H. Cheng. Automatically generating behavioral models of adaptive systems to address uncertainty. In *MoDELS '08: Proceedings of the 11th international conference on Model Driven Engineering Languages and Systems*, pages 568–583, Berlin, Heidelberg, 2008. Springer-Verlag.

[8] A. Gounaris, C. Yfoulis, R. Sakellariou, and M. Dikaiakos. Robust runtime optimization of data transfer in queries over web services. *Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on*, pages 596–605, April 2008.

[9] Helmut, W. Roman, and H. K. Anna. Distributed energy efficiency in future home environments. *Annals of Telecommunications*, 63(9-10):473–485, 8 2008.

[10] M. Izquierdo, J. Jimenez, and A. del Sol. Matlab software to determine the saving in parallel pumps optimal operation systems, by using variable speed. In *Energy 2030 Conference, 2008. ENERGY 2008. IEEE*, pages 1–8, Nov. 2008.

[11] R. Jabr, A. Coonick, and B. Cory. A homogeneous linear programming algorithm for the security constrained economic dispatch problem. *Power Systems, IEEE Transactions on*, 15(3):930–936, Aug 2000.

[12] F. Javed and N. Arshad. On the use of linear programming in optimizing energy costs. In *IWSOS '08: Proceedings of the 3rd International Workshop on Self-Organizing Systems*, pages 305–310, Berlin, Heidelberg, 2008. Springer-Verlag.

[13] F. Javed and N. Arshad. Adopt: An adaptive optimization framework for large-scale power distribution systems. *Self-Adaptive and Self-Organizing Systems, International Conference on*, 0:254–264, 2009.

[14] F. Javed and N. Arshad. A penny saved is a penny earned: Applying optimization techniques to power management. In *16th IEEE International Conference on the Engineering of Computer-Based Systems (ECBS 2009), 13-16 April 2009, San Francisco, CA, USA*, 2009.

[15] B. Khargharia, S. Hariri, F. Szidarovszky, M. Houri, H. El-Rewini, S. Khan, I. Ahmad, and M. Yousif. Autonomic power & performance management for large-scale data centers. *Parallel and Distributed Processing Symposium, 2007. IPDPS 2007. IEEE International*, pages 1–8, March 2007.

[16] A. Kuhn and T. Verwaest. Fame, a polyglot library for metamodeling at runtime. *Models@Runtime 2008*, 2008.

[17] C. Lefurgy, X. Wang, and M. Ware. Server-level power control. *Autonomic Computing, 2007. ICAC '07. Fourth International Conference on*, pages 4–4, 11-15 June 2007.

[18] G. Mainland, D. C. Parkes, and M. Welsh. Decentralized, adaptive resource allocation for sensor networks. In *NSDI'05: Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation*, pages 315–328, Berkeley, CA, USA, 2005. USENIX Association.

[19] R. Nathuji, C. Isci, and E. Gorbatov. Exploiting platform heterogeneity for power efficient data centers. *Autonomic Computing, 2007. ICAC '07. Fourth International Conference on*, pages 5–5, 11-15 June 2007.

[20] E. Ogston, A. Zeman, M. Prokopenko, and G. James. Clustering distributed energy resources for large-scale demand management. In *SASO '07: Proceedings of the First International Conference on Self-Adaptive and Self-Organizing Systems*, pages 97–108, Washington, DC, USA, 2007. IEEE Computer Society.

[21] B. Pickering, S. Robert, S. Menoret, and E. Mengusoglu. Model-driven management of complex systems. In *Workshop on Models@Runtime 2008*, 2008.

[22] K. Shah and M. Kumar. Distributed independent reinforcement learning (dirl) approach to resource management in wireless sensor networks. *Mobile Adhoc and Sensor Systems, 2007. MASS 2007. IEEE Internatonal Conference on*, pages 1–9, Oct. 2007.

[23] M. Sánchez, I. Barrero, J. Villalobos, and D. Deridder. An execution platform for extensible runtime models. In *Workshop on Models@Runtime 2008*, 2008.

[24] M. Wang, N. Kandasamy, A. Guez, and M. Kam. Adaptive performance control of computing systems via distributed cooperative control: Application to power management in computing clusters. *Autonomic Computing, 2006. ICAC '06. IEEE International Conference on*, pages 165–174, 13-16 June 2006.

[25] X. Wang, Y.-H. Song, and Q. Lu. A coordinated real-time optimal dispatch method for unbundled electricity markets. *Power Systems, IEEE Transactions on*, 17(2):482–490, May 2002.

[26] X. Zhu, D. Young, B. Watson, Z. Wang, J. Rolia, S. Singhal, B. McKee, C. Hyser, D. Gmach, R. Gardner, T. Christian, and L. Cherkasova. 1000 islands: Integratedcapacity and workload management for the next generation data center. In *Autonomic Computing, 2008. ICAC '08. International Conference on*, pages 172–181, June 2008.