Systematic Evaluation of Design Decisions in Case-Based Reasoning Systems*

Juan Carlos Santamaría and Ashwin Ram
College of Computing
Georgia Institute of Technology
Atlanta, Georgia 30332-0280
{carlos,ashwin}@cc.gatech.edu

1 Introduction

Two important goals in the evaluation of artificial intelligence (AI) systems are to assess the merit of alternative design decisions in the performance of an implemented computer system and to analyze the impact in the performance when the system faces problem domains with different characteristics (Aha, 1992; Cohen & Howe, 1989; Kibler & Langley, 1988). Achieving these objectives enables us to understand the behavior of the system in terms of the theory and design of the computational model, to select the best system configuration for a given domain, and to predict how the system will behave when the characteristics of the domain or problem change. In addition, for case-based reasoning and other machine learning systems, it is important to evaluate the improvement in the performance of the system with experience (or with learning), to show that this improvement is statistically significant, to show that the variability in performance decreases with experience (convergence), and to analyze the impact of the design decisions on this improvement in performance. This is particularly difficult in implementations of case-based reasoning (CBR) systems because such systems are typically very complex, as are the tasks and domains in which they operate. However, such an evaluation is essential if one is to go beyond mere empirical observation to the prediction and explanation of empirical results.

In this chapter, we present a methodology for the evaluation of CBR and other AI systems through systematic empirical experimentation over a range of system configurations and environmental conditions, coupled with rigorous statistical analysis of the results of the experiments. This methodology enables

*To appear in Leake, D.B., editor (in press). *Case-Based Reasoning: Experiences, Lessons, and Future Directions.* AAAI Press.

us to understand the behavior of the system in terms of the design decisions incorporated into the computational model, to select the best system configuration for a given domain, and to predict how the system will behave in response to changing domain and problem characteristics as it gains experience.

We illustrate this methodology with a case study in which we evaluate a multistrategy case-based and reinforcement learning system which performs autonomous robotic navigation. In this case study, we evaluate a range of design decisions that are important in CBR systems, including *configuration parameters* of the system (e.g., overall size of the case library, size or extent of the individual cases), *problem characteristics* (e.g., problem difficulty), *knowledge representation decisions* (e.g., choice of representational primitives or vocabulary), *algorithmic decisions* (e.g., choice of adaptation method), and *amount of prior experience* (e.g., learning or training). We show how our methodology can be used to evaluate the impact of these decisions on the performance of the system and, in turn, to make the appropriate choices for a given problem domain and verify that the system does behave as predicted.

2 Background

Two important characteristics of CBR systems are that they are complex and the task domains in which they operate are also complex and often ill-structured. As a result, the behavior of a CBR system has many sources for variability which causes any performance measure defined to evaluate this behavior to have variability as well. This in turn makes it difficult to assess the significance of an observed behavior of the system in a specific situation. Similarly, due to the complexity of the system and problem domains, theoretical analysis of the system performance given alternative decisions and domain characteristics, although desirable, is difficult in many cases (Francis & Ram, 1995; Kibler & Langley, 1988). However, straightforward performance curves that show how the performance of a system improves over time are not good enough. Although these curves show that the performance improves on specific test problems, they do not provide useful information about why the system works or how specific design decisions affect the behavior of the system, nor can they be used to predict the behavior of the system under different circumstances.

Ablation or lesion studies can be used to analyze the impact of different system's modules in the performance of the system (Cohen & Howe, 1988; Kibler & Langley, 1988). In such studies, one or more system modules are removed or deactivated to analyze how the performance of the system changes. Although these studies do provide some information about the contribution of different modules to the overall performance of the system, they are based on extreme operating conditions that are often impractical (i.e., one or more modules are set to be either active or inactive). Moreover, design decisions often deal with allocating certain amount of resources to different modules. Due to their nature,

ablation studies can only deal with all-or-nothing resource allocation, and cannot be used to investigate resource or other interactions between modules.

As an alternative to these approaches, empirical model fitting and statistical tools can be used to analyze the change in the performance of the system in terms of alternative decisions and domain characteristics. In such an analysis, the system is evaluated through systematic experiments designed to test the impact of various design decisions while filtering out undesirable sources of variability. The results of the experiments are then analyzed using statistical tools. In addition, in CBR systems and other systems that learn from experience, it is important to verify not only that the performance of the system improves with experience, but also that the variability in the performance of the system when solving new similar problems decreases as the number of problems solved in the past increases. Systematic empirical analysis based on statistical tools can also be used to verify the significance of these behaviors and to identify the sources of variability in the behavior of the system.

This chapter presents an evaluation methodology based on well-known statistical tools that can be used to explicitly analyze the merit of a range of design decisions in the performance of a system. The methodology consists of designing experiments to carefully control the variability in the behavior of the system and to obtain empirical performance data over a range of alternative design parameters, representational choices, algorithmic choices, and domain characteristics. This data is used to construct a mathematical model that relates the change in the performance of the system to design decisions and domain characteristics. The model can be used to select the best system configuration for a given domain, to predict the behavior of the system when the domain characteristics change, and to show how (or whether) performance improves with experience.

This chapter is organized as follows. Section 3 describes the proposed methodology. Section 4 illustrates the methodology via a case study in which the methodology is used to evaluate SINS, a case-based system that performs autonomous robotic navigation. The case study focuses specifically on design decisions that are relevant in CBR systems, although the methodology proposed here can be used for evaluating a wide range of AI systems. Section 5 concludes by summarizing the implications of the methodology for the evaluation of CBR and other AI systems.

3 Evaluation Methodology

The proposed evaluation method is shown in Table 1. It consists of five phases: experimental design, data collection, model construction, model validation, and robustness analysis. During the experimental design phase, the *factors* that may influence the performance of the system are identified. These factors are classified in two broad groups: *design decisions* and *domain characteristics*. Experiments are designed to measure the performance of the system while sys-

Table 1: Systematic evaluation methodology.

1.	Experimental Design
2.	Data Collection.
3.	Model Construction.
4.	Model Validation.
5.	Robustness Analysis (including Learning Profiles).

tematically varying the factors. In the data collection phase, the experiments are executed and data is collected. During the model construction phase, empirical *models* that relate the design decisions, domain characteristics, and the performance of the system are constructed. In the model validation phase, the assumptions identified during the model construction phase are verified. In this manner, the models can be used to state valid conclusions about the relationship between the system's performance and the factors (i.e., design decisions and domain characteristics). During the robustness analysis phase, the system is tested under different alternatives for the factors to assess the generality of the results. In addition, for a CBR or machine learning system, learning profile curves are constructed to compare the empirical models with traditional learning curves obtained with the system. These profiles further verify the results and also provide an intuitive feel for the changes in the external performance of the system as it gains experience. Finally, the data collection and analysis at each step may raise new questions and new ideas for continuing the evaluation, perhaps with a different focus or using a different set of factors; this results in iterative use of the methodology until the issues of interest are resolved. The following sections discuss each of these phases in more detail.

3.1 Experimental Design and Data Collection

Case-based reasoning systems are typically complex in nature and, like other AI systems, their performance depends on several factors. These factors can broadly be classified into two categories: design decisions and domain characteristics. **Design decision factors** are related to the configuration of the system and often deal with allocating resources to different modules within the system. In CBR systems, several factors are of interest: *system configuration parameters*, such as the size of the case library, thresholds for quality of match or distance metrics for retrieval, and other numerical parameters (such as learning rates in machine learning systems); *knowledge representation factors*, such as the choice of representational primitives or the content theory of the domain; and *algorithmic factors*, such as the specific algorithms used for retrieval or adaptation. **Domain characteristic factors** are related to problem descriptions and are used to categorize problems in the domain: for example, the difficulty of the

problems or the variability in the kinds of problems that the system might face. An additional factor that is relevant to system performance on a given problem is the *amount of prior experience* that the system has had. If it is feasible to train the system before it is deployed or to hand-code the appropriate experiences, the amount of prior experience can be viewed as another design decision factor for the system designer to consider.

To understand and optimize the performance of the system, it is necessary to assess the role of each factor in the system's overall behavior. During the first phase of our evaluation methodology, factors of interest are identified and experiments are designed to measure the system's performance for different alternatives or *levels*[1] for each factor. A representative sample of system configurations and problem instances is selected, each one with a different set of alternatives for each factor. Several versions of the system are built, each with a different set of choices for the design parameters, representations, and algorithms.

An *experiment* consists of measuring the performance of each version of the system executing on each set of problem instances. Thus, an experiment requires more than a single run; it requires several runs carried out under different conditions (i.e., different configurations of the system, different representational vocabularies, different algorithms for each module, different environmental configurations, different levels of problem difficulty, different order of problems, etc.). Data is gathered from all runs and analyzed using statistical techniques. This allows us to determine not only which factors influence the behavior of the system but under which circumstances and to what extent. (These relationships are captured mathematically in the model construction phase, discussed later.) This information can be used both to understand why the system worked and to select the best system configuration for a given problem domain.

While designing the experiments, it is important to reduce unwanted sources of variability in the system's performance across runs. This ensures that the empirical model attributes differences in system performance solely to differences between alternative chosen factors to the extent possible. If there are other sources of variability in the performance that were not considered beforehand, the model will lose its accuracy and will not be useful in estimating the best design alternatives for optimum system performance. Unwanted sources of variability usually originate by not considering relevant factors in the model or by having a poor experimental design. For example, if the selected sample of problem instances have different levels of difficulty and if the level of difficulty is not considered as a factor, then a poor experimental design might be to measure the performance of two systems configured with a different set of alternatives on only one set of problem instances. If one version of the system solves the easier problems while the other solves the more difficult ones, it will be difficult to assess whether the difference in system performance is due to the different

[1] The term *treatment* is also used to denote a level.

system configurations or to different difficulty levels. To avoid this problem, the experiments should either *balance* out the runs along the factors (i.e., run all system configurations on problems instances with all levels of difficulty), or *block* out the runs along a specific factor (i.e., run all system configurations on problem instances with only one level of difficulty).

The choice about when to balance or to block a specific factor is made by trading off the cost of running experiments against the range of applicability of the results of the empirical model to be constructed. A model is applicable only to the range of problem instances from which it was constructed. Increasing the range of problem instances increases the range of applicability of the model but also increases the number of experiments needed because each version of the system must be run with problem instances that represent the entire range. Due to the fact that factors are often grouped by design decisions and domain characteristics, one practical way to design the experiments is to balance out all the factors related to design decisions and to block out all the factors related to domain characteristics. In this way, a detailed analysis of the merits of the design decisions under specific but representative problem instances can be obtained. Such an analysis allows the selection of the best alternatives for the design decisions considered so that the system's performance is optimum when working under the representative problem instances. Next, during the robustness analysis phase, the generality of the selected "best" system configuration can be studied across different levels of domain characteristic factors.

An approach similar to this use of the robustness analysis phase is described by Aha (1992). He proposes an evaluation methodology designed to understand the effect of different domain characteristics in the performance of learning systems and to derive rules that designers can use to decide when to generalize the results obtained from case studies. In contrast, our methodology is designed to understand the effects of the design decisions on the performance of the system, to determine if the results are significant, and furthermore, to analyze the domain characteristics under which the evaluation study is valid. Thus, our methodology helps us evaluate the theoretical basis for the system and provides a means to select appropriate design parameters for a given application.

At the end of the experimental design phase, the data collection phase is executed. This simply involves running the different versions of the systems on the chosen problem sets and gathering performance data using appropriate performance metrics.

3.2 Model Construction

After the experiments are run and the data collected, a mathematical model is constructed to fit and explain the results. Models that relate system performance and relevant factors (i.e., design decisions and domain characteristics) are useful because they provide information about how each factor influences the performance of the system. Such models can serve many purposes, such as

predicting what the performance of the system would be under certain conditions of interest, and selecting appropriate system parameters, representations, and algorithms to configure a system for specific situations. The model can also help explain the behavior of the system and thereby provide insight into the theory underlying the design of the system.

Due to the complexity of case-based reasoning systems and the often ill-structured task domains that they operate in, theoretical models that relate system performance and relevant factors are difficult to construct. Instead, the data collected during experiments can be used to infer an empirical model. Empirical models are mathematical expressions based on experimental data and can be constructed using statistical estimation techniques. The basic idea in constructing a model is to assume that there exists a functional relationship between system performance and the relevant factors. The model is a mathematical expression of this relationship.

An example of a linear empirical model is shown in Equation 1.

$$\mathrm{Y_i} = \beta_0 + \beta_1 \mathrm{X_{1i}} + \cdots + \beta_\mathrm{k} \mathrm{X_{ki}} + \varepsilon_\mathrm{i} \tag{1}$$

In equation 1, the results of $i = 1, \ldots, n$ experimental runs are assumed to follow the additive relationship expressed above. In the model, Y_i represents the dependent variable or observed performance of the system for each of the n runs, and the X_{ji} represent the independent variable or alternative values of each of the $j = 1, \ldots, k$ factors for each of the n runs. Note that dependent variables (also known as response variables) may be used to measure not only the external performance of the system according to some metric but also internal behaviors or effects that are of interest. The values ε_i represent the *residual* or error incurred by the model in estimating the observed value Y_i given the values of the X_{ji} for each of the runs. Inferential statistical techniques are used to estimate the values of the β_j coefficients for the given sample.

This empirical model is applicable to a wide range of situations. The response variable Y_i must be a numerical quantity representing either an observed or qualitative variable that measures the performance of the system. For example, for a robot that navigates autonomously, an observed response variable may be the time the robot takes to reach the destination point, whereas a qualitative response variable may be the "smoothness" of the trajectory of the robot on a scale from 1 to 10 as determined by some judge. The independent variables X_{ji} are numerical quantities representing the alternative values of the factors under consideration. These variables may be continuous or categorical. *Continuous variables* represent factors that can be varied continuously, such as the amount of memory given to the system under study (e.g., the maximum size of the case library) or a numerical threshold parameter (e.g., the minimum value of a quality of match metric that is necessary to allow a case to be retrieved). *Categorical variables* represent discrete alternatives, such as different representational vocabularies, different domain theories, or different adaptation algorithms that can be used in various modules of the system.

A widely used inferential statistical technique is regression analysis using least-squared error (LSE) (Neter *et al*, 1989). LSE estimates the model coefficients (the β_j's in equation 1) by minimizing the squared sum of the residuals (ε_i) across the sample. The model coefficients represent the influence of each factor ($X_j i$) on the performance of the system (Y_i). LSE estimation is widely used because of its properties; in particular, the Gauss-Markov theorem establishes that LSE estimators for regression models are the best linear unbiased estimators possible (Weld, 1916).

The linear regression model in Equation 1 is very general and can incorporate a wide range of smooth functional relations. For example, ablation studies analyze the partial increment/decrement in the system's performance with the addition/elimination of a system component (Cohen & Howe, 1988). Such analyses can be performed using a regression model in which categorical variables can take on the values 1 or 0 to indicate whether a system component is present or not. A linear regression model can also be used with continuous valued parameters, such as amount of memory. Finally, quadratic terms or other functional forms can be incorporated into the model because the only restriction is that it must be linear in the β_j coefficients. Thus, the model can capture mathematically the impact of individual design decisions on the performance of the system as well as that of interactions between design decisions. Once the model is created, the best set of parameter values can be selected to optimize the performance of the system. When smooth functional relations do not apply, either the domain of the model is too ample and must be reduced or more terms (e.g., quadratic, cubic, etc.) must be added into the model. This follows from the fact that any functional relationship can be approximated by a series of polynomials with any degree of accuracy as the number of terms of the polynomials increases (Munroe, 1963).

A common problem when constructing a model is selecting appropriate independent variables to use from the set of all the possible variables that might be considered. Selecting all possible independent variables in the model may artificially reduce the error between the data and model due to overfitting. One solution to this problem is to consider all the possible subsets of independent variables and select the best model according to a specific criterion. One criterion that is often used is to select the model with the best adjusted multiple coefficient of determination (R^2_{adj}). This coefficient measures the ability of the model to explain the variability of the response variable (Y_i). The greater the R^2_{adj}, the better the model explains the variability of the response variable in terms of the variability of the independent variables. The adjusted multiple coefficient of variation takes into consideration the number of terms (estimated parameters) in the model in such a way that models that include terms to artificially reduce the error have lower R^2_{adj} values.

3.3 Model Validation

Any model estimated using an inferential statistical technique relies on a set of assumptions. The validity of the model constructed depends on the extent to which these assumptions hold for a given sample of data. For example, there are two assumptions implied in the additive model of Equation 1. First, the residuals are assumed to have zero mean and constant variance across samples. Second, they are assumed to be independent and normally distributed. When these assumptions do not hold, conclusions derived from the model may not be valid. Deviations from the assumption of the residuals having constant variance might lead to overestimates in the ranges of parameter values. This in turn would cause the model to be inaccurate. Small deviations from the assumption of the residuals being normally distributed usually does not create serious problems, but major departures are of concern since the conclusions derived using the model might be incorrect.

To verify qualitatively that the residuals have constant variance, a plot of the residuals against each independent variable and against the fitted response variable is used. A normal probability plot is commonly used to verify the normality distribution assumption of the residuals.

3.4 Robustness Analysis

In the final phase of the methodology, alternative levels for the factors are tried and verified against the model. As suggested in the experimental design phase, the experiments in this phase should focus on the factors that are associated with the domain characteristics. In this way it is possible to analyze the sensitivity of the best system configuration as obtained from the model across different domain characteristics, and to verify the generality of the system across a range of problems.

In addition, for a CBR or machine learning system, learning profiles are constructed to further verify the empirical model and to provide an intuitive look at the internal behaviors and external performance of the system as it gains experience. A learning profile is similar to a traditional learning curve, except that it also plots the predicted performance of the system based on the model in order to compare that with the actual performance.

4 A Case Study

This section describes a case study based on an implemented case-based reasoning system and is intended as an example of how to apply the methodology proposed in the previous section. The case study is based on a detailed and systematic analysis of a system that performs autonomous robotic navigation. The objective of this evaluation is twofold: first, to find a model that describes

the relationship between the system's configuration parameters, design decisions, experience level, and its performance as measured by a suitable metric; and second, to evaluate the robustness of the performance of the system under different environmental conditions. The first objective enables us to understand the relationship between the configuration parameters, design decisions, and system performance, to evaluate the merit of alternative design decisions in the behavior of the system, and to verify that the performance of the system continues to improve with large amounts of experience. Moreover, a model that relates the performance metric with the configuration parameters and amount of experience is also useful because it enables us to pick the best system configuration parameters for a given situation and to determine the amount of prior experience (or training) that is necessary to achieve a desired level of performance. The second objective, evaluation of the robustness of the system when performing under different environmental conditions, is useful because it enables us to verify the robustness and generality of the system, that is, whether it is likely that the results obtained will hold when the system runs under different environments.

The following subsections describe in more detail the system we used in this evaluation and how each step in the evaluation methodology was carried out. For the purposes of this chapter, we will focus on a representative set of design decisions that are relevant in many case-based reasoning systems.

4.1 System Description

SINS (Self-Improving Navigation System) is a case-based reasoning system that performs autonomous robotic navigation in unstructured terrains (for a detailed technical discussion of the system, see Ram & Santamaría, 1993a; for a discussion of the case-based reasoning aspects of the system, see Ram & Santamaría, 1993b). Autonomous robotic navigation is defined as the task of autonomously and safely moving a robot from a source point to a destination point in an obstacle-ridden terrain. SINS uses reactive control methods for navigation coupled with case-based reasoning methods for adaptation and learning during task performance. SINS is implemented on a Denning MRV-III robot and can be used to drive the actual robot as well as a simulation.

It is difficult to evaluate a system such as SINS because its behavior is the result of many factors interacting with each other and because it is designed to work under unstructured terrains. Also, some modules in the architecture perform random actions under certain conditions (e.g., to accomplish exploration behaviors). This causes the evaluation to be even more difficult because random actions increase the variability in the behavior of the system. Thus, as discussed earlier, measuring the performance of the system during a single run (or an ad hoc set of runs) or performing ablation studies does not accomplish the objectives of a systematic evaluation, which are to analyze the impact of the design decisions and domain characteristics on the performance of the system and to

select appropriate design parameters for a given application. As discussed below, a systematic statistical evaluation using our methodology can be used to accomplish these objectives.

Briefly, SINS consists of a navigation module, which uses a schema-based reactive control method (Arkin, 1989), and an on-line adaptation and learning module, which uses case-based reasoning and reinforcement learning methods (Ram & Santamaría, 1993a). The navigation module is responsible for moving the robot through the terrain from the starting location to the desired goal location while avoiding obstacles along the way. A set of control parameters can be used to change the behavior of the navigation module. The adaptation and learning module is responsible for learning control parameters to change the behavior of the navigation module in such a way that the performance of the navigation task is improved. In particular, the adaptation and learning module constructs mappings from sensory input information about the environment to appropriate control parameters that should be used in that environment. These mappings are represented as "cases" that encapsulate the system's navigational experiences.

A case in SINS represents continuous sensory inputs and associated motor schema control parameters over a time window, and recommends appropriate control parameters to use in different situations. Cases are matched not only on the basis of the current instantaneous situation but on the immediate history of situations over a suitable time window. As the system gains experience, it can create new cases by allocating unused memory or it can modify previous cases by updating their content or by increasing their time windows (which changes their temporal extent). Several design decisions affect the behavior and performance of SINS; in this case study, we focus on four that are relevant to the case-based reasoning component of the system. Two of these are system configuration parameters: the maximum number of cases (C) and the maximum case size in terms of the maximum time window over which experiences can be encapsulated (S). These parameters together determine the maximum amount of memory the system can use to store its experiences. When the maximums are reached, the system can use new experiences only to modify the content of the cases if it is appropriate to do so.

The other two design decisions evaluated in this study are categorical decisions: the choice of representational primitives, and the choice of adaptation algorithms. For the former, there are several choices available for the sensory inputs that the learning and adaptation module uses to represent environmental situations. One option is to use sensory inputs with "low-level" information; in the navigation domain, this might involve using an ultrasonic sensors to measure the distance of the closest obstacle in the direction of the goal. Another option is to use sensory inputs that encode "high-level" information; for example, using an array of ultrasonic sensors to compute a measure of the density of obstacles surrounding the robot. The former type of sensory input is very specific and may allow SINS to discover the best control parameters the robot

may use in specific situations. However, we would also expect that the cases learned by the system to be very specific and not work as well in similar but not identical situations. In contrast, the latter type of sensory input is more generic and may allow SINS to discover good control parameters. However, the learned cases may turn so coarse that the system may not perform near-optimally in most cases.

Another design decision in SINS is the choice of the adaptation algorithm. Every few steps, as determined by a configuration parameter, the adaptation and learning module may recommend new control parameters to the navigation module. To accomplish this, it retrieves the case most similar to the current environmental situation over a recent time window and adapts control parameters used by the navigation module based on the control parameters suggested by the case. The best values for the control parameters would be those that increase the likehood of obtaining a positive outcome in the next cycle (as measured by performance metrics such as progress towards the goal or number of collisions with obstacles). In order to learn appropriate control parameters, SINS must "explore" the space of possible parameter values, that is, it must try several values for each environmental situation and learn which values produce positive or negative results. This presents at least two design options for the adaptation method. One option is to select the new values of the control parameters stochastically according to the outcome they have achieved in the past. In this method, values that have produced positive outcomes (or obtained positive rewards) are more likely to be selected than those that have produced negative outcomes. Another option is to select for each control parameter the value that has produced the best reward in the past in any situation. The former adaptation method enforces more exploration and may discover better solutions than the later one. On the other hand, due to the same reason, the latter method may converge faster than the former one.

The performance of SINS depends not only on its design but also on the problem environments in which it is placed, that is, on the domain characteristics. SINS navigates in randomly generated environments consisting of rectangular bounded worlds. Each environment contains circular obstacles, a start location, and a destination location. The location, number, and radius of the obstacles are randomly determined to create environments of varying amounts of *clutter*, defined as the ratio of free space to occupied space. 15% clutter corresponds to relatively easy environments and 25% clutter to difficult environments. A randomly generated navigation problem in a high clutter world is likely to be more difficult than one in a low clutter world. Thus, an important factor to consider from the domain characteristic group is the clutter value of the environments in which the robot is placed. Finally, since SINS is a learning system, its performance also depends on the amount of previous experience it has had.

To summarize, the performance of SINS varies due to design decisions (system configuration parameters, representational vocabulary, adaptation algorithm), domain or problem characteristics (world clutter), and amount of ex-

perience. Moreover, due to the nature of the task and the architecture of the system, SINS can perform differently given the same world and case library. The reason for this is twofold: first, the navigation module includes a behavior which generates random motion to allow the robot to get out of local minima; and second, the adaptation and learning module tunes the navigation module randomly when no appropriate case exists so as to allow the system to explore and discover new regularities. This means that any performance metric used to evaluate the system must be treated as a random variable and statistical estimation techniques should be used to assess its mean value.

The following sections describe two empirical studies. The first study focuses on the categorical design decisions (representational and algorithmic choices), and the second focuses on the continuous design decisions (system configuration parameters) and domain characteristics (problem difficulty). Both studies evaluate improvement in the performance of the system with experience.

4.2 Study 1: Choice of Input Representation and Adaptation Method

4.2.1 Experimental Design and Data Collection

The objective of this study is to find an empirical model that describes the relationship between the choice of input representation, adaptation method, and system performance as well as the conditions under the model is applicable. In this study, we evaluated two choices of input representations and two choices of adaptation methods. The input representation refers to the information that the learning and adaptation module of SINS uses to represent and categorize the surrounding environment (situation). The adaptation method refers to the methods by which this module adapts the control parameters suggested by the best matching case to the current situation.

The two choices of input representations studied were the low-level and high-level information types introduced earlier. The low-level information type consists of the following four sensory inputs: **Obstacle-Distance-Ahead**, **Obstacle-Distance-Behind**, **Obstacle-Distance-Right**, and **Obstacle-Distance-Left**. Each of these variables provide a measure of the nearest obstacle that impede navigation in the direction towards, contrary to, right of, and left of the direction of the perceived goal respectively. The high-level information type consists of the following four sensory inputs: **Obstacle-Density** provides a measure of the occupied areas around the robot that impede navigation[2]; **Absolute-Motion** measures the activity of the system; **Relative-Motion** represents the change in motion activity over an appropriate interval; and **Motion-Towards-Goal** specifies how much progress the system has actually made towards the goal. Both types of sensory inputs are computed and constantly updated using the

[2]Note that this sensory input does not provide any information about the distances or direction of the obstacles; it simply measures the density of occupied area around the robot.

information received from the robot's physical sensors (in our case, 24 ultrasonic sensors arranged around the robot every 15 degrees and shaft encoders).

The two choices of adaptation method studied were the stochastic method and the select-best method. The stochastic adaptation algorithm selects control parameter values randomly but favoring to values that lead to positive reward. Specifically, values are selected according to a Boltzmann distribution: $P(v) = \frac{exp(w_v)}{\sum_i exp(w_i)}$, where the w_i represent the expected reward for value i. The select-best adaptation method simply selects the value that led to the most positive expected reward in the past (i.e., the value with largest w_v).

To collect data for model construction and analysis, we performed several simulated runs on the system. A run consisted of placing the robot at the start location and letting it run until it reached the destination location. The data for the estimators was obtained after the system terminated each run. This was to ensure that we were consistently measuring the effect of learning across experiences rather than within a single experience.

We evaluated the performance of SINS using the median value of the time it takes to solve a world. The reason for this is that the median is a robust estimator of the mean and is not too sensitive to outliers. Outliers are common in schema-based reactive control since the system can get trapped in local minima points, resulting in a significant change in the behavior of the system. An experiment consisted of measuring the time SINS takes to solve a world across five independent runs under the same conditions (i.e., same conditions but different random seeds) and reporting the median among the five runs as the response variable.

Table 2 shows the design matrix for the study. It consists of a 2×2 factorial design: two choices for input representation (low-level and high-level), and two choices for adaptation method (stochastic and best). Starting from no prior experience, every setup was exposed to thirty levels of experience, each level being one complete navigation problem. We recorded the outcome of five replications for every level of experience. Thus, the total number of system runs in study was $2 \times 2 \times 30 \times 5 = 600$. The domain characteristics in this experiment were blocked by evaluating all the setups under the same randomly generated 10% cluttered world. This allowed us to balance out the effects of the design decision choices and experience level (which were of interest) and block out the effects of other parameters such as world clutter.[3]

4.3 Model Construction

The performance of SINS was evaluated by estimating the median time to solve a world. Thus, the model that needed to be determined in this study has the

[3] The choice of the problem world is merely for illustrative purposes and is arbitrary as far as this chapter is concerned. The methodology can be also used to evaluate system performance on a range of problems instead of a single representative problem.

Table 2: Experimental design matrix.

Choice of input representation	high-level / low-level
Choice of adaptation method	stochastic / best-value
Experience Levels	1-30
World Clutter	10%
Replicates	5
Response Variable	Time
Total runs	600

median time (T) as the response variable, and would relate T with the choices of input representation and adaptation method, and with amount of experience. Note that a lower value of T corresponds to better performance on the task. We used the following regressors as independent variables: input representation (I; low=0, high=1), adaptation method (A; stochastic=0, best-value=1), and experience level (E). We also considered additional regressors, such as the interaction terms IE, AE, and IA. The reason for considering all these factors is to allow for the possibility that interaction terms may explain variability in the response variable better than individual terms. Statistical analysis was used to reveal which of these terms were really significant and should be considered in the final model. Equation 2 shows the complete hypothetical model.

$$\begin{aligned} T &= \beta_0 + \beta_I I + \beta_A A + \beta_E E + \\ &\quad \beta_{IE} IE + \beta_{AE} AE + \beta_{IA} IA + \beta_{IAE} IAE + \\ &\quad \varepsilon \end{aligned} \tag{2}$$

Figure 1 shows the response variable T collected during the experiment. The figure shows that there are mainly two phases as experience increases: a learning phase, where performance improves; and a maturity phase, where performance remains constant. (This is as intuitively expected; recall that SINS is learning to navigate a fixed world because domain characteristics have been blocked in this study.) A single first-order or second-order polynomial would not be able to fit all the data with enough accuracy in both phases. As described before, there are two ways to solve this problem: either increase the number of terms in the model, or reduce the region of operability of the model. Since there are clearly two phases as experience increases, we decided to use two models, one for each phase.

Table 3 shows the alternative empirical models based on the design decision choices. The terms β_I and β_A measure the change in the intercept of the equation given a change in the choice of input representation and adaptation method, respectively. The term β_{IA} measures the interaction effect of the design decisions on the intercept. The terms β_{IE}, β_{AE}, and β_{IAE} perform the same

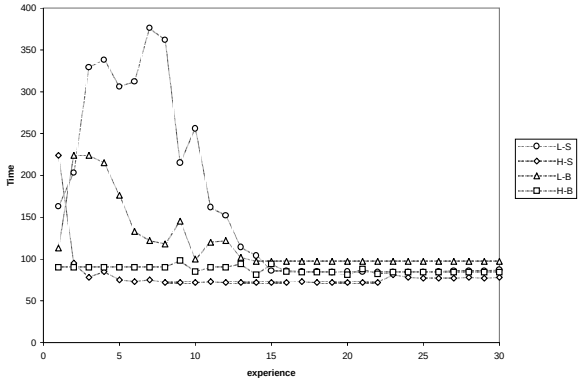


Figure 1: Response variable vs. experience level. The four graphs correspond to low-level representation with stochastic adaptation (L-S), high-level representation with stochastic adaptation (H-S), low-level representation with best-value adaptation (L-B), and high-level representation with best-value adaptation (H-B).

Table 3: Design for Experiment 1.

Input representation	Adaptation method	Equation
low-level	stochastic	$T = \beta_0 + \beta_E E$
high-level	stochastic	$\bar{T} = (\beta_0 + \beta_I) + (\beta_E + \beta_{IE})E$
low-level	best-value	$\bar{T} = (\beta_0 + \beta_A) + (\beta_E + \beta_{AE})E$
high-level	best-value	$\bar{T} = (\beta_0 + \beta_A + \beta_{IA}) + (\beta_E + \beta_{AE} + \beta_{IAE})E$

role but on the slope of the equation. In this context, the slope represents the learning rate of the system, that is, how much the system improves performance (decrease in T) per unit of experience (increase in E).

An all-subsets regression analysis was performed to determine which of the terms in the model are really significant, that is, which terms influence the response variable. In this analysis, all possible subsets of regressors are considered and a model is constructed using each subset. Tables 4 and 5 show the results of this analysis of the models for the learning and maturity phases. We measured the optimality of the model by its R^2_{adj}, the adjusted coefficient of multiple determination[4] This coefficient measures the ability of the model

[4]The R^2_{adj} is the R^2 value adjusted to take into account the number of parameters in the model. This allows models having different numbers of parameters to be compared.

Table 4: Best subsets regression results for the learning phase.

Number of variables	R^2_{adj}	$\hat{\sigma}$	I	A	E	IE	AE	IA	IAE
1	35.3	66.14	X						
2	45.6	60.64	X		X				
3	52.8	56.48	X				X		X
4	64.0	49.33	X	X	X				X
5	64.5	48.69	X	X	X	X		X	
6	65.9	47.99	X	X	X	X	X	X	
7	65.3	48.43	X	X	X	X	X	X	X

Table 5: Best subsets regression results for the maturity phase.

Number of variables	R^2_{adj}	$\hat{\sigma}$	I	A	E	IE	AE	IA	IAE
1	50.9	5.61	X						
2	94.1	1.95	X	X					
3	95.0	1.79	X	X				X	
4	96.5	1.50	X	X		X			X
5	97.1	1.38	X	X		X		X	X
6	97.0	1.38	X	X	X	X		X	X
7	97.0	1.39	X	X	X	X	X	X	X

to explain changes in the response variable by changes in the regressors. Its range is between 0%, which means that none of the variation in the response is explained by variation in the regressors, and 100% which means that all of the variation in the response is explained by variation in the regressors. Thus, the larger the R^2_{adj} the more explicative is the model. This coefficient also indicates if we are overfitting the data by introducing more terms that artificially reduce the error in fit between the model and the data.

Tables 4 and 5 show the best model obtained within each subset of constant size or number of variables. In these tables, $\hat{\sigma}$ is the estimated standard deviation of the dependent variable in the model, and R^2_{adj} is the adjusted coefficient of multiple determination as discussed earlier. The "X's" show which variables are included in the best model for each size.

During the learning phase, the best model obtained with the all-subset analysis corresponds to the one having six regressors as independent variables ($F = 20.04$, P-value $= 0.000$).[5] There is no statistical evidence that the term β_{IAE} is significant, that is, there is no evidence that the learning rate is affected

[5]The F statistic is used to determine the significance of the regression. The P-value is the probability determined by F; the lower this value the better the result, since the significance of the regression is $(1 - \text{P-value})\%$ which is 100% for P-value $= 0$.

Table 6: Model coefficients for the learning phase.

Coefficients	Value	Std. Error	90% C.I.
β_0	334.19	23.420	(294.99,373.39)
β_E	−12.79	3.484	(−16.95,−8.63)
β_I	−209.56	28.870	(−257.89,−161.23)
β_A	−132.43	28.870	(−180.78,−84.10)
β_{IE}	7.90	2.868	(3.09,12.70)
β_{AE}	5.14	2.868	(0.34,9.94)
β_{IA}	96.00	24.780	(54.51,137.49)

by the interaction between the choice of input representation and adaptation method. Table 6 shows the statistical results for each individual parameter in the model for the learning phase as well as the 90% confidence interval estimate of their real values.

The results show that the system configuration that achieves best performance is the one using the high-level representation and the stochastic adaptation algorithm ($T = 51.73$ seconds at $E = 15$ experiences). However, the fastest learning rate corresponds to the system configuration with the low-level representation and the stochastic adaptation algorithm ($\beta_E = -12.79 \frac{\text{secs.}}{\text{exp.}}$) since changing the level of input representation or the adaptation method reduces the learning rate (increases the slope). There is a big influence on the intercept (performance with no previous experience) caused by the design decisions. The interaction term (β_{IA}) shows that the amount of change in performance caused by changing one of the design decisions depends on the other design decision. For example, the average change in performance caused by switching from low-level to high-level input representation when using the stochastic adaptation method is $\beta_I = -209.56$ seconds. But, if the same switch is performed using the best-value adaptation method, the average change in performance is only $\beta_I + \beta_{IA} = -113.56$ seconds. In both cases, it appears that using high-level input representations provides a better start in the learning process. There is no statistical evidence ($t = 0.68$, P-value $= 0.499$) to indicate that a change in the input representation improves the learning rate more than a change in the adaptation method (i.e., $\beta_{IE} = \beta_{AE}$).

During the maturity phase, the best model obtained with the all-subset analysis corresponds to the one having five regressors as independent variables ($F = 389.99$, P-value $= 0.000$). There is no statistical evidence that the terms β_E and β_{AE} are significant. This means that there is no evidence that there is further learning going on after experience level 15. There is some indication that performance varies with experience in the high-level representation/stochastic adaptation system configuration, but it is positive and small ($\beta_{IE} = 0.53$ seconds/experiences). Table 7 shows the statistical results for each

Table 7: Model coefficients for the maturity phase.

Coefficients	Value	Std. Error	95% C.I.
β_0	85.07	0.355	(84.47,85.66)
β_I	-14.07	1.956	$(-15.45,-12.68)$
β_A	11.93	0.502	(11.09,12.77)
β_{IE}	0.53	0.082	(0.39,0.66)
β_{IA}	9.27	2.766	(4.64,13.90)
β_{IAE}	-0.54	0.116	$(-0.73,-0.34)$

individual parameter in the model for the maturity phase as well as the 90% confidence interval estimation of the real coefficient values.

The results confirm the hypothesis that the system is in a maturity phase since two terms related with experience are not in the model and the others are statistically significant but with small values. Again, the best system configuration is the one using high-level input representations and the stochastic adaptation method ($T = 73$ seconds). Furthermore, there is an interaction effect in the design decisions on the final performance of the system. For example, the average change in performance caused by switching from low-level to high-level input representation when using the stochastic adaptation method is $\beta_I = -14.07$ seconds. But if the same switch is performed using the best-value adaptation method, the average change in performance is only $\beta_I + \beta_{IA} = -12.67$ seconds. In both cases, the use of high-level input representations provides better performance after convergence. There is statistical evidence that the best-value adaptation procedure causes the final performance to deteriorate by an average of $\beta_A = 11.93$ when the system uses low-level input representations and by an average of $\beta_A + \beta_{IA} = 21.20$ when the system uses high-level input representations. This confirms the hypothesis that the stochastic adaptation method takes longer to converge but arrives at better solutions.

4.3.1 Model Validation

The above results demonstrate the power of our systematic evaluation method: the method enables us to obtain an in-depth understanding of the performance of the system and to make design decisions in a principled manner. However, there are two assumptions that must be verified before accepting the proposed models (and associated results) as a valid: the residuals have zero mean and constant variance, and the residuals have normal distribution. The LSE technique relies on these assumptions; thus, since the model coefficients were calculated using this technique we must verify if these assumptions hold. Otherwise, any conclusions derived from the model could be wrong. In particular, violation of the assumption about the residuals having zero mean and constant variance

could introduce inaccuracy in the estimation of the model coefficients, and violations of the assumption of the residuals being normally distributed could produce underestimation of the confidence intervals.

A scatter plot of the residuals against the fitted response was used to diagnose changes in variance, and a normal probability plot of the residuals was used to verify the normality distribution of the residuals. The results of each of these two validation techniques are shown in Figures 2 and 3 for the learning and maturity phases, respectively. The left chart in Figure 2 shows a systematic diagonal pattern that appears for the lower values of the fitted response. Usually, the presence of this pattern indicates that additional terms may be required, but in this study this is to be expected since the response is not linear (refer to Figure 1). For the purposes of this chapter, we decided to accept this deficiency due to the complications of introducing more terms into the model. The left chart in Figure 3 shows a constant band of residuals distributed randomly along the horizontal axis. Visually, this appears as a set of dots distributed evenly and randomly across the figure from left to right. When the variability of the residuals is not constant, the band tends to narrow or widen along the horizontal axis. Since this is not the case, we infer that the variance is indeed constant, as is necessary for the model to be valid.

The right charts of Figures 2 and 3 show normal probability plots of the residuals for the learning and maturity phases. In these charts, the values of the residuals are plotted against their expected value as drawn from a normal distribution. If the residuals are indeed normal, then the plots should show a straight line that crosses the origin. In both models, the residuals have zero mean but there is some departure from linearity, especially in the model corresponding to the maturity phase. The reason for this is that the variance in the maturity phase was very small since the system responded similarly during the five replications at each experience level.

Since the two assumptions, residuals with zero mean, constant variance, and normally distributed are not fully satisfied, care must be taken in generalize this result to other 10% cluttered worlds. One option at this point would be to repeat the analysis with additional terms, as discussed above. Another option is to repeat the experiments, this time systematically varying the world clutter; this is discussed next.

4.3.2 Robustness Analysis

A follow-up experiment were designed to evaluate the generality of the system by running experiments with a different set of problems. Since the previous experiments indicated that the best design for the system was to use high-level input representations and the stochastic adaptation algorithm, the system was configured in this manner for the new experiment. The data for the experiment was collected in the same manner as the previous experiment, but with the difference that the system solved a fixed randomly-generated 7%-cluttered world

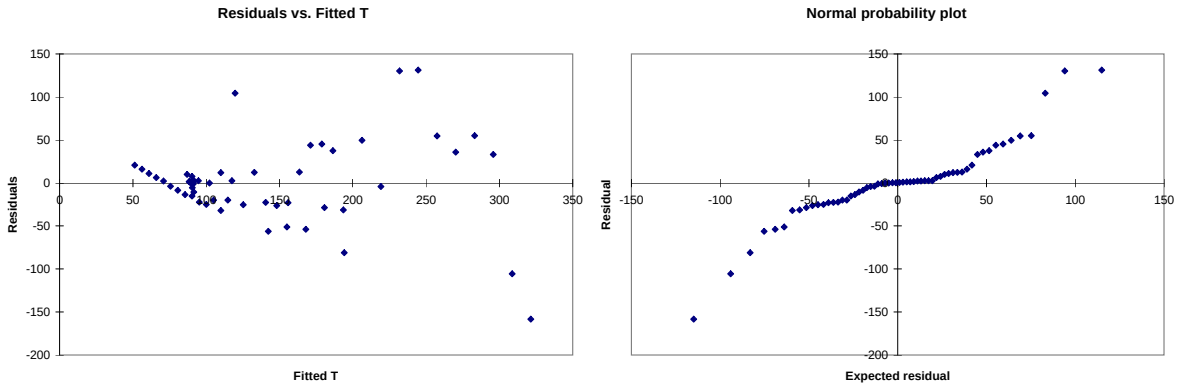


Figure 2: Residual plots for the learning phase.

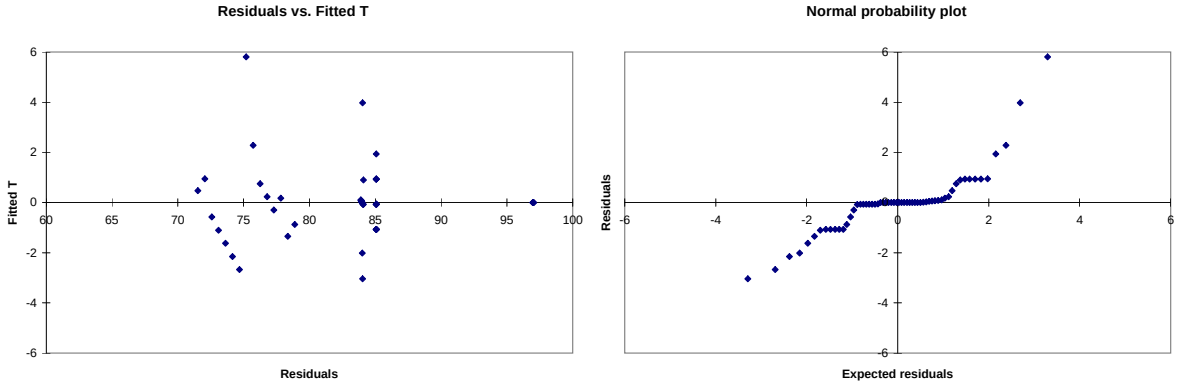


Figure 3: Residual plots for the maturity phase.

Table 8: Experimental design matrix.

Choice of input representation	high-level
Choice of adaptation procedure	stochastic
Experience Levels	1-30
World Clutter	7%
Replicates	5
Response Variable	Time
Total runs	150

Table 9: Model coefficients for the learning phase.

Coefficients	Value	Std. Error	90% C.I.
α_0	102.97	2.76	(97.46,108.48)
α_E	-0.95	0.30	$(-1.56,-0.35)$

in every run.

As in the previous experiment, the model that needs to be determined has the median time (T) as the response variable. But in this case, the model relates the response variable with the amount of experience only since the other factors are constant. If such a model is found to be significant (i.e., the model shows that the amount of experience is related to the response variable), we can conclude that the system learns under the new environmental conditions. Furthermore, the coefficient derived from this model can be compared with the respective coefficient in the previous model. If a significant difference is detected, we can conclude that changing the world clutterness from 10% to 7% affects the learning performance. Table 8 shows the experimental design matrix and Equation 3 shows the complete hypothetical model for the this experiment.

$$\mathrm{T} = \alpha_0 - \alpha_\mathrm{E}\mathrm{E} + \varepsilon \tag{3}$$

This model is a simplification of the model in equation 2 where the experience level (E) is the only regressor. Tables 9 and 10 show the statistical results for each individual parameter in the model as well as the 90% confidence interval estimation of its value for the learning and maturity phases, respectively.

The results show that during the learning phase, the learning rate decreased from $-(\beta_E+\beta_{IE}) = 4.89$ seconds/experience to $-\alpha_E = 0.96$ seconds/experience. This means that the system decreases the learning rate when solving a simpler world. Additionally, the value of the intercept reduced from $\beta_0 + \beta I = 124.64$ seconds to $\alpha_0 = 102.97$ seconds. This confirms that that the system is indeed solving a simpler world and most of the learning occurs during the early expe-

Table 10: Model coefficients for the maturity phase.

Coefficients	Value	Std. Error	90% C.I.
α_0	92.71	0.56	(91.59,93.84)

riences. During the maturity phase, the term α_E is not statistically significant. This means that the system does not improve the performance further after experience level 15, which confirms the hypothesis that it is in the maturity phase.

4.4 Study 2: Maximum Number of Cases, Maximum Case Size, and World Clutter

4.4.1 Experimental Design and Data Collection

The second study focuses on continuous design decisions (specifically, numerical parameters) as opposed to categorical ones. The objective of this study is to find an empirical model that describes the relationship between relevant system configuration parameters and its performance as well as the conditions under which the model is applicable. The configuration parameters studied were the maximum number of cases (C) and maximum case size (S). The first parameter limits the number of different cases the system may store in its case library, and the second parameter limits the amount of information a single case can hold. To block the effects of the categorical design decisions, we chose the low-level input representation and the stochastic adaptation method which were kept constant in these experiments.

The data collection in this study was similar to that in the previous study. We chose a randomly-generated 15% cluttered performance world on which the system was to be tested.[6] We performed several runs on the system, where a run consisted of placing the robot at the start location and letting it run until it reached the destination location. As in the previous study, the performance of the system was evaluated using the median value of the time it takes to solve a world. We ran different system configurations, each configured using different C and S parameters. The set of runs obtained with each configuration was replicated five times. This allowed us to collect the data required to build a model that relates the system performance with the configuration parameters and amount of experience.

Table 11 shows the design matrix of the experiment. It consists of a 4×4

[6] As before, the choice of the problem is arbitrary and merely for illustrative purposes. If desired, additional test problems can be introduced and the system tested on each individually or in sequence, although in the latter case additional terms may need to be introduced into the equations in order to account for variability in system performance across problems.

Table 11: Experimental design matrix.

Maximum number of cases	15, 25, 35, 45
Maximum case size	6, 9, 12, 15
Experience Levels	1-30
World Clutter	10%
Replicates	5
Response Variable	Time
Total runs	2400

factorial design: four levels for maximum number of cases, and four levels for maximum case size. Each setup was exposed to thirty levels of experience, and the outcome of five replications for every level of experience was recorded. Thus, the total number of experiments in this study was $4 \times 4 \times 30 \times 5 = 2400$.

4.4.2 Model Construction

The performance of SINS was evaluated by estimating the median time to solve a world. Thus, the model that needs to be determined for this study has the median time (T) as the response variable; the model relates T with the configuration parameters and amount of experience. A fundamental difference between this model and the one used in the previous study is that the regressors are continuous instead of categorical. This means that the model can be used to predict system performance using values that are not the ones that were used during the experiment. This is an advantage of continuous regressors over categorical regressors. We used the following regressors as independent variables: maximum number of cases (C), maximum case size (S), and amount of experience (E). We also considered additional regressors such as the quadratic terms C^2, S^2, and E^2 and the quadratic interactions CE, SE, and CS. The reason for considering all these factors is to allow for the possibility that interaction terms may explain variability in the response variable better than individual terms. Statistical analysis was used to reveal which of these terms are really significant and should be considered in the final model. Equation 4 shows the complete hypothetical model.

$$\begin{aligned} T \quad = \quad & \beta_0 + \beta_C C' + \beta_S S' + \beta_E E' + \\ & \beta_{CS} C' S' + \beta_{CE} C' E' + \beta_{SE} S' E' + \\ & \beta_{CC} C'^2 + \beta_{SS} S'^2 + \beta_{EE} E'^2 + \\ & \varepsilon \end{aligned} \tag{4}$$

where: V' is the standardized value[7] of a variable V (i.e., $V' = \frac{V - \bar{V}}{\sqrt{\mathrm{var}(V)}}$).

[7]Use of standardized values instead of the original values helps to reduce roundoff errors and other problems with multicollinearity between independent variables.

Table 12: Best subsets regression results.

Number of variables	R^2_{adj}	$\hat{\sigma}$	C	S	E	CS	CE	SE	C^2	S^2	E^2
1	53.8	11.031			X						
2	66.5	9.394	X		X						
3	73.6	8.346	X		X		X				
4	75.7	8.002	X		X		X		X		
5	77.3	7.732	X		X		X		X		X
6	78.8	7.482	X	X	X		X		X		X
7	79.0	7.434	X	X	X		X	X	X		X
8	79.2	7.402	X	X	X		X	X	X	X	X
9	79.4	7.372	X	X	X	X	X	X	X	X	X

Assuming that the mathematical relationship between the response variable and the independent variables is "smooth", a second order polynomial expression of that relationship, such as the one proposed by the model, is a good approximation. The quadratic terms for the maximum number of cases and maximum case size allowed for the possibility of utility problems, and the interaction terms were included to allow for the possibility of a direct relationship between the response variable and the terms.[8]

An all-subsets regression analysis was performed to determine which of the terms in the model are significant. Table 12 shows the results of this analysis. The best model obtained with the all-subset analysis corresponds to the one having all the regressors as independent variables ($F = 205.824$, P-value $= 0.000$).[9] Table 13 shows the statistical results for each individual parameter in the model as well as the 95% confidence interval estimation of its real value.

Table 14 shows the analysis of variance (ANOVA). The ANOVA table is a statistical tool that it is used to determine the sources of variability in a model. The first column identifies a source of variation, and the second, third, and fourth columns show the degrees of freedom (df), sum of squares (SS), and mean squared (MS) of a source, respectively. The fifth column shows the value of the F statistic which is used to determine the significance of the regression. The sixth column shows the P-value. A high significance value means that the variation in the response variable is indeed explained by variation of the independent variables or regressors.

Considering this model, the optimal system configuration parameters can be found using standard calculus techniques, i.e., by setting the first partial deriva-

[8] Among the three interaction terms only CS has physical meaning. The interaction term CS is a direct measure of the total amount of memory available to the system. This is an example of a particularly difficult evaluation problem since different design decisions can influence each other under conditions of resource limitations.

[9] As before, the F statistic was used to determine the significance of the regression.

Table 13: Model coefficients.

Coefficients	Value	Std. Error	95% C.I.
β_0	72.23	0.78	(70.70,73.77)
β_E	-11.92	0.34	$(-12.58,-11.26)$
β_C	-5.79	0.34	$(-6.45,-5.13)$
β_S	1.97	0.34	(1.31,2.63)
β_{EE}	2.33	0.38	(1.59,3.07)
β_{CC}	2.99	0.42	(2.16,3.82)
β_{SS}	-0.95	0.42	$(-1.78,-0.12)$
β_{CE}	-4.32	0.34	$(-4.99,-3.66)$
β_{SE}	-0.91	0.34	$(-1.57,-0.24)$
β_{CS}	0.74	0.34	(0.08,1.41)

Table 14: ANOVA table.

Source	df	SS	MS	F	P-value
Regression	9	100675.7	11186.2	205.824	7.1E-157
Residual	470	25543.7	54.3		
Total	479	126219.4			

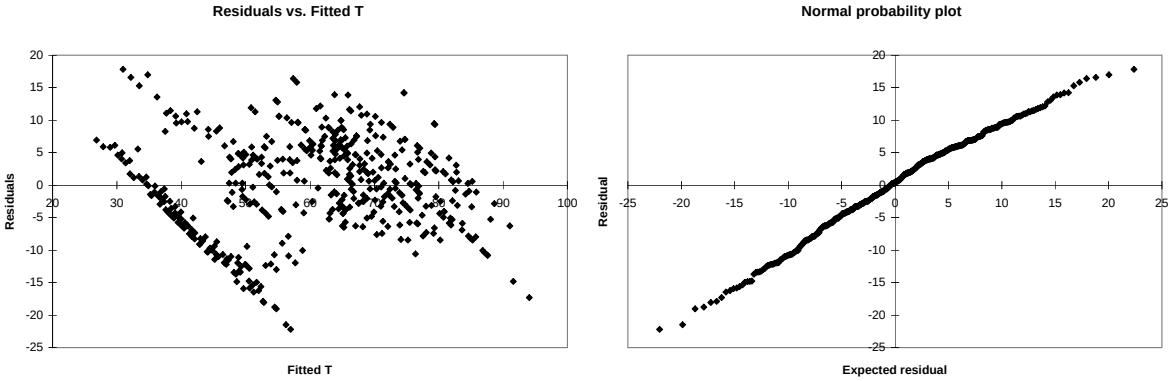
tives of the model with respect the relevant parameters to zero. Equations 5 and 6 show the optimal values for C' and S' at a given level of experience E'.

$$\begin{aligned} C' &= \frac{2\beta_{SS}\beta_C - \beta_{CS}\beta_S}{\beta_{CS}^2 - 4\beta_{CC}\beta_{SS}} + \frac{2\beta_{SS}\beta_{CE} - \beta_{CS}\beta_{SE}}{\beta_{CS}^2 - 4\beta_{CC}\beta_{SS}} E' \\ &= 0.80 + 0.75E' \end{aligned} \tag{5}$$

$$\begin{aligned} S' &= \frac{2\beta_{SS}\beta_S - \beta_{CS}\beta_C}{\beta_{CS}^2 - 4\beta_{CC}\beta_{SS}} + \frac{2\beta_{SS}\beta_{SE} - \beta_{CS}\beta_{CE}}{\beta_{CS}^2 - 4\beta_{CC}\beta_{SS}} E' \\ &= 0.05 + 0.41E' \end{aligned} \tag{6}$$

According to these equations the optimal configuration parameter values change with the level of experience. This is due to the interaction terms that exist among those variables. These equations can be used to determine the optimum configuration of the system for a given situation. For example, if the system were going to receive 20 training problems, we might want to optimize the performance of the system at experience level $E = 20$ (i.e., $E' = 0.52$). In this case, we would configure the system using 43 maximum cases ($C' = 1.19$) of size 11 ($S' = 0.26$).

Figure 4: Residual Plots.



4.4.3 Model Validation

As in the first study, a scatter plot of the residuals against the fitted response was used to diagnose changes in variance and a normal probability plot of the residuals was used to verify the normality distribution of the residuals. The results of each of these two validation techniques are shown in Figure 4. The left chart in Figure 4 shows a constant band of residuals along the horizontal axis. Thus, this chart indicates that the variability of the residuals is constant along the fitted values of the response variable (i.e., median time), which is necessary for a valid model. However, as in the previous study, some diagonal patterns are present. This indicates that a second order polynomial does not perfectly characterize the behavior of the system at certain levels of performance. The values at which this happens corresponds to the values where the system has reached maturity and there is no further learning.

The right chart in Figure 4 shows a normal probability plot of the residuals. In this chart, the values of the residuals are plotted against their expected value as drawn from a normal distribution. If the residuals are indeed normal, the plot should show a straight line that crosses the origin. Since this is actually the case, we can conclude that the residuals are normal. Since the two assumptions, residuals with zero mean and constant variance and residuals having normal distribution, hold, the model can be considered valid.

4.4.4 Robustness Analysis

Finally, as before, a followup experiment was designed for the robustness analysis phase of the methodology to evaluate the generality of the SINS approach. In this experiment, we evaluated a specific configuration of the system performing under a different environment. The data for the experiment was collected in the same manner as the first experiment, the only difference being that the agent

Table 15: Model coefficients.

Coefficients	Value	Std. Error	95% C.I.
α_0	80.2	0.71	(78.57,81.47)
α_E	-2.86	0.48	(-3.84,-1.87)
α_{EE}	2.53	0.55	(1.41,3.65)

solved a fixed randomly-generated 20%-cluttered world in every run. The configuration parameters for the system were selected using the model constructed in the first experiment and to optimize the performance of the system around an experience level E equal to 20, that is, with 43 maximum cases of size 11, as determined earlier.

As in the previous experiments, the model that needs to be determined has the median time (T) as the response variable. But in this case, the model only relates the response variable with the amount of experience since the other factors are constant. The coefficient derived from this model can be compared with the respective coefficients derived in the previous model. If a significant difference is detected, we can conclude that changing the world clutterness from 15% to 20% affects the learning performance. Equation 7 shows the complete hypothetical model for the second experiment. This model is a simplification of the model in equation 4 where only the experience level (E') is included in the regressors. Table 15 shows the statistical results for each individual parameter in the model as well as the 95% confidence interval estimation of its value.

$$\mathrm{T} = \alpha_0 - \alpha_{\mathrm{E}}\mathrm{E}' + \alpha_{\mathrm{EE}}\mathrm{E}'^2 + \varepsilon \tag{7}$$

As the model shows, a bigger intercept value is obtained which means that the system indeed needs more time to solve a 20% cluttered world. Also, the increased world clutter has a big influence in the rate of learning (-α_E), which is reduced from $-(\beta_E + \beta_{CE}C' + \beta_{SE}S') = 17.30$ seconds/experience to $-\alpha_E = 2.86$ seconds/experience. This means that experience does not improve system performance as quickly as for the 15% cluttered world or, in other words, the learning rate is higher in the easier world. The acceleration of the learning rate (α_{EE}) does not seem to be influenced by the change of world clutter (i.e., it is in the 95% confident interval of β_{EE}).

4.4.5 Learning profiles

While the above results demonstrate the effectiveness of the SINS approach, it is also interesting to look at the learning profile of the system, if only to provide an intuitive feel for changes in the internal behaviors and external performance of the system as it gains experience. The learning profile of the system can be determined by comparing the model's predictions with traditional learning

curves based on the system's actual performance. This is shown in Figure 5. The graphs labeled "Median" show the actual time taken by the system (starting with no prior experience or hand-coded high-level knowledge) on a randomly generated 15% cluttered world; each point in the graph is the median of five replications with the system configured using the optimal parameters as derived in the previous section. The graphs labeled "Model" show the predicted performance from empirical models. The graph on the left describes the general empirical model along with the error bars for that model. The general model is the performance model derived using the data obtained using the entire range of configurations of the design parameters (Equation 4 and Table 13). The graph on the right describes the performance model for the specific configuration used in these runs; this was obtained by redoing the statistical analysis for the data obtained from this configuration alone.

As can be observed from the figure, the performance of the system falls within the general and specific models. The general model has bigger error bars because it must fit the performance of the system over the entire range of configurations. The specific model fits the data better but can only be used to predict the system's performance with this particular system configuration. The results demonstrate that the system does indeed improve its performance with experience, and that this improvement is as predicted by the empirical models derived from the statistical analysis. Furthermore, the performance of the system approaches its optimal level around $E = 20$, which is as expected because, as discussed previously, the configuration parameters used for these experiments were explicitly chosen to optimize performance at that experience level. This provides further evidence for the validity of the analysis.

4.5 Discussion

In order to illustrate the kinds of conclusions that can be drawn from this evaluation methodology, let us briefly discuss the above results in the context of the SINS system. The performance of SINS is very complex and depends not only on simple terms but also on their interactions. Study 1 shows that high-level input representations favor learning and enable better solutions than low-level representations. Additionally, study 1 shows that the system using the stochastic adaptation method takes a longer time to converge than the best-value adaptation method; however, once converged, the former arrives at better solutions than the latter. There is no interaction between the choice of input representations and the adaptation procedures. Further study is necessary to find out the effect of the choice of input representation and the knowledge acquired in one world on the performance of the system while solving a different world.

Study 2 shows that the median time taken by the system (when configured with low-level input representations and the stochastic adaptation procedure) to solve a 15% cluttered world decreases mainly with the experience level. Increas-

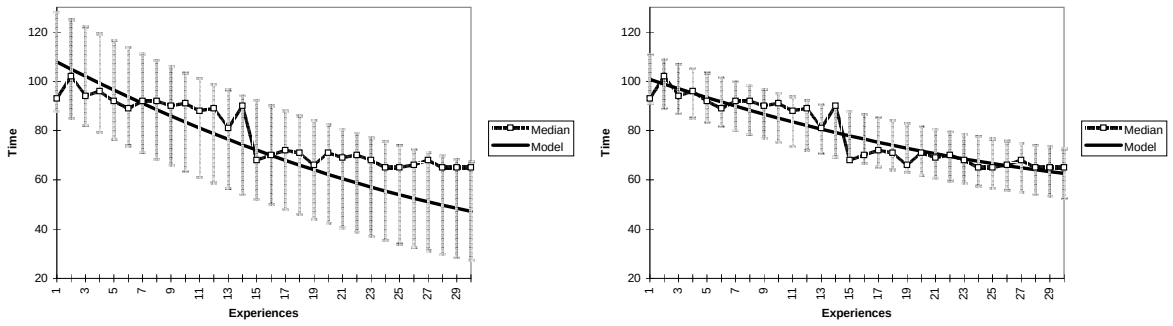


Figure 5: Performance profiles showing median time from actual system runs and predicted time using two empirical models. On the left is the general model constructed using the data from the entire experiment, i.e., over the entire range of system configurations. On the right is the specific model constructed using only the data from the specific configuration used in the system runs.

ing the maximum number of cases also improves the performance, but a positive coefficient in its quadratic term may harm the performance for big values. On the other hand, the maximum case size has a positive linear coefficient and a negative quadratic coefficient which indicate that large cases may improve performance as compared to small cases. Negative interaction coefficients indicate that for bigger values of maximum number of cases and case size, the system requires more experiences to start improving its performance. Intuitively, this is to be expected since the more space available to store regularities, the more experience level is required to avoid overfitting and to construct reliable regularities. Finally, the performance of SINS is influenced by the world clutter, the learning rate being the factor subject to the greatest influence.

In summary, the evaluation was useful to verify and understand several aspects of SINS. In particular:

- The evaluation showed that SINS does improve its performance significantly with experience (Tables 6 and 13).
- Study 1 showed that the choice of input representation and the choice of adaptation method independently influence the performance of SINS and that there are no interactions between these decisions (Table 6).
- Study 1 showed that high-level input representations provide a better starting point for learning than low-level input representations. Also, it shows that the stochastic adaptation method takes longer to converge than

the best-value adaptation method, but once it does, it arrives at better solutions (Tables 6 and 7).

- Study 1 showed that using the regressors (I, A, and E) and their interactions the empirical model can only account for 65.3% and 97.0% of the variability in the performance of the system during the learning and maturity phases, respectively. This result is important because it captures the limitations of the derived model. Part of the remaining percentage could be explained by introducing more factors or by changing the functional forms of the terms in Equation 2; the rest of the variation in performance is due to the randomness in the system.

- Study 2 showed that the performance of the system in a 15% cluttered world depends on the selected system configuration parameters, as well as on interactions among them (Equation 4 and Table 13).

- Study 2 showed the best way to configure SINS in a 15% cluttered world for a prespecified level of experience (Equations 5 and 6).

- Studies 1 and 2 showed how a change in the environment characteristics, namely cutter, affected the performance of SINS (Equations 3 and 7).

- Study 2 showed that using the proposed factors (C, S, and E) and their interactions the empirical model can only account for 79.8% (i.e., R^2=0.798) of the variability in the performance of the system. Part of the remaining 21.2% could be explained by introducing more factors or by changing the functional forms of the terms in Equation 4; the rest of the variation in performance is due to the randomness in the system.

5 Conclusions

Case-based reasoning systems are typically very complex, and the behavior and performance characteristics of such systems are the result of many interacting factors that originate from the many design decisions that go into building them. These design decisions range from numerical system parameters to the knowledge representation scheme chosen to represent the domain and the algorithms and metrics used for matching, adaptation, and other functional modules of the system. Additionally, the tasks, domains, and problems that these systems have typically addressed are also very complex and have a significant influence on the behavior and performance of the system. A good evaluation must show not only that a system is performing well; it should also inform us about the performance of the system under various conditions and about learning or improvement in performance with experience. Furthermore, the evaluation must provide insight about how various design decisions influence both performance and learning. Such an evaluation provides an in-depth understanding of the behavior of the

system. It allows the researcher to analyze the theory or computational model based on empirical experiments with the computer program, and the system designer and end user to optimize the configuration of the computer program for a given situation of interest. A better understanding of the behavior of the system across domain characteristics also allows the designers to predict the performance of the system under different situations and to determine the conditions under which the system will perform adequately.

In this chapter we proposed a systematic evaluation methodology which enables us to obtain these evaluation objectives for implemented computer systems. The methodology involves identifying the design decisions to be analyzed and designing system experiments which systematically exercise the computer program over the range of design options and problem domains of interest. Based on the experimental results, a mathematical model is constructed which can be used to derive statistically significant conclusions and provide the desired insights into the behavior of the system. The methodology can be used to evaluate CBR systems as well as a broad range of machine learning and other AI systems.

6 Acknowledgments

The authors wish to thank Mary Dowling for her comments and suggestions regarding the statistical analysis for the case study, and David Aha, Anthony Francis, Todd Griffith, Mimi Recker, and anonymous reviewers for their comments on an earlier draft of this chapter.

7 References

Aha, D.W. 1992. Generalizing from Case Studies: A Case Study. In *Proceedings of the Ninth International Conference of Machine Learning*, 1-10, Aberdeen, Scotland.

Arkin, R.C. 1989. Motor schema-based mobile robot navigation. *International Journal of Robotics Research*, (84):92-112.

Bareiss, R. 1989. The Experimental Evaluation of A Case-Based Learning Apprentice. In *Proceedings of the Case-Based Reasoning Workshop*, 162-167, Florida.

Cohen, P.R. 1989. Evaluation and Case-Based Reasoning. In *Proceedings of the Case-Based Reasoning Workshop*, 168-172, Florida.

Cohen, P.R. & Howe, A.E. 1988. How evaluation guides AI research. *AI Magazine*, 9(4):35-43.

Cohen, P.R. & Howe, A.E. 1989. Towards AI research Methodology: Three Case Studies in Evaluation. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(3):634-646.

Francis, A.G. & Ram, A. 1995. A Comparative Utility Analysis of Case-Based Reasoning and Control-Rule Learning Systems, *Eighth European Conference on Machine Learning (ECML-95)*, Crete, Greece.

Kibler D. & Langley, P. 1988. Machine Learning as an Experimental Science. In *Proceedings of the Third European Working Session on Learning*, 81-92, Glasgow, UK.

Koton, P. 1989. Evaluating Case-Based Problem Solving. In *Proceedings of the Case-Based Reasoning Workshop*, 173-175, Florida.

Lehner, P.E. 1989. Toward an Empirical Approach to Evaluating the Knowledge Base of an Expert System. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(3):658-662.

Munroe, M.E. 1963. *Modern Multidimensional Calculus*. Addison-Wesley, Massachussetts, 1963.

Neter, J., Wasserman, W., Kutner, M.H. 1989. *Applied Regression Models*.

Minton, S. 1990. Quantitative Results Concerning the Utility of Explanation-Based Learning. *Artificial Intelligence*, 42:363-391.

Ram, A. & Santamaría, J.C. 1993a. Multistrategy Learning in Reactive Control Systems for Autonomous Robotic Navigation. *Informatica*, 17(4):347-369.

Ram, A. & Santamaría, J.C. 1993b. Continuous Case-Based Reasoning, in *Proceedings of the AAAI Workshop on Case-Based Reasoning*, 86–93, AAAI Press.

Weld, L. 1916. *Theory of Errors and Least Squares*. New York, Macmillan.