2004-01-01

# Guided Model Checking with a Bayesian Meta-heuristic

Michael D. Jones
mike.jones@byu.edu

Peter Lamborn

Kevin Seppi
*Brigham Young University*, kseppi@byu.edu



## Original Publication Citation

Kevin Seppi, Michael Jones, and Peter Lamborn. "Guided Model Checking with a Bayesian Meta-heuristic." In Fundamenta Informaticae, 7(1-2), 26, pp. 111-126.

# Guided Model Checking with a Bayesian Meta-heuristic

Kevin Seppi, Michael Jones and Peter Lamborn
Brigham Young University
Department of Computer Science
3328 TMCB, Provo UT, 84602, USA
kseppi@cs.byu.edu, jones@cs.byu.edu, pcl@cs.byu.edu

## Abstract

*This paper presents a formal verification algorithm for finding errors in models of complex concurrent systems. The algorithm improves explicit guided model checking by applying the Empirical Bayes method to revise heuristic estimates of the distance from a given state to an error state. Guided search using the revised estimates finds errors with less search effort than the original estimates.*

## 1 Introduction

Concurrency increases the complexity of a protocol, which increases the probability that a protocol contains an error. Model checking is a verification technique that can establish both the presence and absence of errors. When used in conjunction with other verification techniques, model checking is most useful for finding errors that occur infrequently after long sequences of unique state transitions. These kinds of errors often arise in concurrent protocols. As a result, model checking, and particularly explicit state enumeration model checking, has become a viable tool for protocol design and verification.

The work presented in this paper increases the error-finding capacity of enumeration model checkers by combining a well-known statistical method with guided model checking. In guided explicit state enumeration model checking, a heuristic guides the search toward regions of the transition graph more likely to contain errors. Given a state, the heuristic function estimates the cost of reaching an error by estimating the number of transitions needed to reach an error. This value is used to determine the order in which states will be expanded. Guided model checkers can find errors more quickly in incorrect designs and can achieve more useful partial coverage in resource-bound problems involving large designs.

Generally, heuristics are treated as point estimates (i.e., a single numeric value). This paper takes a slightly different interpretation of heuristic values. We treat heuristic values as random variables (i.e., functions that assign a real valued probability between 0 and 1 to each possible outcome of an event) [1, 2]. A probability density function (pdf) is used to characterize the distribution of inaccuracy in the heuristic. If the heuristic is an accurate estimate, then most of the probability will be close to the actual distance to the target. Interpreting the heuristic as a random variable allows us to assess and improve the quality of the heuristic using statistical methods. For the purpose of this paper, we use mean squared error to measure the quality of a heuristic.

Our Bayes heuristic search algorithm ("BHS") minimizes mean squared error using an Empirical Bayes [3, 4] meta-heuristic. In this paper, a meta-heuristic is a function which takes heuristic values as input and returns improved heuristic values (with reduced mean squared error). Our Empirical Bayes meta-heuristic uses estimates from a set of sibling states to derive the confidence that should be attributed to each individual estimate. The confidence level is then used to proportionally revise the original estimate toward the mean of the sibling estimates. Estimates with low confidence are revised more severely than estimates with high confidence. We validate this approach theoretically using a Bayesian model and show that the resulting heuristic values have smaller total expected mean squared error.

We give experimental results in which the BHS algorithm finds errors in the same or fewer states as conventional heuristic search in 17 of 22 problems. Conventional heuristic search is a best-first search using an inadmissible (i.e., may overestimate the cost of reaching a target) property-dependent heuristic. Efficiency is measured by the number of states explored before reaching an error state. In 7 of 22 problems, BHS explored at least one order of magnitude fewer states than conventional heuristic search.

We also compare the performance of BHS and conventional heuristic search on 100 variants of the same problem. This experiment measures the robustness of BHS on a family of related problems. The BHS algorithm found errors in 95 problem instances after exploring up to 32,500 states per

Proceedings of the Fourth International Conference on Application of Concurrency to System Design (ACSD'04)
0-7695-2077-4/04 $ 20.00 © 2004 IEEE

problem instance. Conventional heuristic search needed to explore up to 249,600 states per instance to solve the same number of instances. The results indicate that, for this problem, the performance improvement obtained by BHS is insensitive to the particular configuration of the search problem.

In the next section, we describe various heuristics that have been used in guided model checking and give a variation of a conventional heuristic which we will use as input to the BHS meta-heuristic. While we phrase our claims in terms of the conventional heuristic function presented in Section 2, we emphasize that BHS can be applied to a wide variety of heuristics which assign costs to states. Section 2 also gives a statistical model for the conventional heuristic. This model justifies distributional assumptions that make BHS simple to implement and facilitate the proof that BHS reduces expected mean squared error.

Section 3 describes the meta-heuristic and the supporting notation. Section 4 introduces the Bayesian model used by the BHS algorithm, and proves that the resulting heuristic values have lower expected mean squared error. Section 5 contains experimental results. Section 6 gives some concluding remarks and directions for future work.

## 2 Heuristics

The performance of a guided model checker depends critically on the heuristic used to guide the search. In this section, we discuss related work in designing heuristics for guided model checking then give the heuristic used in the rest of the paper. This section concludes with an empirical study of the inaccuracy (variance) observed in our heuristic across a set of different models. An approximate assessment of the inaccuracy in our heuristic is needed for the Bayesian meta-heuristic introduced in the next section.

### 2.1 Related Heuristics for Guided Model Checking

Several kinds of heuristics have been proposed for use in guided model checking. Yang and Dill used a combination of Hamming distance, preimage computation and approximation and user annotations (called guideposts) in guided search [5]. A combination of user annotations and error state preimage computation reduced the number of states explored in 8 of 8 guided search problems. Our BHS does not require user annotations and does not perform preimage computation, but could be used to revise state rankings generated from Hamming distance and user annotations.

Bloem et. al. classify heuristics for guided model checking as system-dependent or property-dependent [6]. Other kinds of heuristics, such as structural-dependent [7] and specification-dependent [8] heuristics have also been proposed. System-dependent heuristics exploit properties of the design under test to reduce memory usage in symbolic model checking algorithms. Property-dependent heuristics estimate the cost of reaching a violation of a property from a given state. Structural-dependent heuristics guide the search based on the structure of the transition graph without reference to the property. Specification-dependent heuristics are like stuctural-dependent heuristics except that specification-dependent heuristics treat the specification as a black box. Treating the specification as a black box forces the heuristic to rank search prioritees based on input/output pairs rather than the structure of the underlying (and hidden) transition graph.

Using a combination of property and system dependent heuristics, Bloem et. al. obtained a reduction in time and space requirements in 6 of 8 reported search problems. Edelkamp et. al. have developed a property-dependent heuristic for use in SPIN, an explicit model checker [9]. This heuristic is the basis for the heuristic defined in Section 2.2. Edelkamp's heuristic is intended to be admissible (i.e. never overestimates the cost of reaching an error) and relies on specific limitations of queue behavior in SPIN to generate better estimates. For example, SPIN queues do not allow reordering so that a message stored $i$ places from the head of a queue can be recieved in no fewer than $i$ transitions. Edelkamp et. al. used best-first search with this heuristic to explore fewer states than depth-first search in 14 of 20 problems.

Although our primary interests in [6] and [9] are the heuristic functions, the primary contributions of [9] and [6] are extensions of guided model checking algorithms to linear temporal logic (LTL) and computational tree logic (CTL) properties, respectively. Our contribution is a new algorithm for computing a meta-heuristic that improves the performance of a heuristic in guided model checking. The meta-heuristic can be applied within the context of any guided model checking algorithm that ranks states using estimated numerical values.

### 2.2 A Property-dependent Heuristic

The empirical results in this paper are obtained using a property-dependent heuristic inspired by Edelkamp [9]. Table 1 contains the heuristic in both inadmissible and admissible forms. The admissible heuristic is included for comparison. Given a state $s$ and a boolean formula, the heuristic $C$ estimates the number of transitions between $s$ and a state that satisfies the formula. In the table, $r$ is a variable quantified over finite domain $R$ for some predicate $p$, $a$ and $b$ are boolean expressions, $x$ and $y$ are arithmetic expressions and $v$ is a boolean variable. Negations are restricted to boolean variables and constants.

| Boolean Formula | Inadmissable | Admissible |
|---|---|---|
| $C(s, \exists r : R.p(r))$ | if$(\exists r : R.p(r))$ then 0 else $avg_{\forall q \in R}(C(s, p(q)))$ | if$(\exists r : R.p(r))$ then 0 else $min_{\forall q \in R}(C(s, p(q)))$ |
| $C(s, \forall r : R.p(r))$ | $\Sigma_{\forall q \in R}(C(s, p(q)))$ | $max_{\forall q \in R}(C(s, p(q)))$ |
| $C(s, a \vee b)$ | if$(a \vee b)$ then 0 else $avg(C(s, a), C(s, b))$ | if$(a \vee b)$ then 0 else $min(C(s, a), C(s, b))$ |
| $C(s, a \wedge b)$ | $C(s, a) + C(s, b)$ | if$(a \wedge b)$ then 0 else $max(C(s, a), C(s, b))$ |
| $C(s, x \oslash y)$ where $\oslash \in (<, >)$ | if$(x \oslash y)$ then 0 else $\mid x - y \mid +1$ | if$(x \oslash y)$ then 0 else 1 |
| $C(s, x \oslash y)$ where $\oslash \in (\geq, \leq, =)$ | if$(x \oslash y)$ then 0 else $\mid x - y \mid$ | if$(x \oslash y)$ then 0 else 1 |
| $C(s, \neg v)$ | if $(v)$ then 1 else 0 | if $(v)$ then 1 else 0 |
| $C(s, v)$ | if$(v)$ then 0 else 1 | if$(v)$ then 0 else 1 |
| $C(s, T)$ | 0 | 0 |
| $C(s, F)$ | $\infty$ | $\infty$ |

**Table 1. Inadmissible and admissible heuristics defined recursively on boolean formulas.**

The admissible heuristic always underestimates the distance between $s$ and a state that satisfies the formula because the heuristic assumes that any predicate can be satisfied in one step. If we make no assumptions about a transition system, then we must assume a formula can be satisfied in one step becuase a transition can move $s$ into a new state $s'$ which satisfies the formulas under test. The inadmissible heuristic relaxes this requirement, and looses admissibility, by making assumptions about the number of transitions needed to satisfy a formula based on values in the formula. These assumptions are reasonable for some transition systems, but not others.

Note that the admissible heuristic presented here differs from the Edelkamp heuristic because it accounts for the possibility of satisfying more than one operand of an operator with a single-step modification of a single variable. This can occur when an expression of the form $a \wedge b$ can be satisfied in a single step. For example, the trivial expression $v_1 \wedge v_1$ is satisfied by setting $v_1$ to "true" in a single step. Such expressions could be re-written to factor out common variables or subexpressions, but for many models, including those used in our tests below, model builders rarely find and remove such redundancy. A similar problem occurs when the values of more than one variable are changed in a single atomic transition (e.g., multiple assignments within atomic transitions in Promela or rules in Murphi).

### 2.3 Properties of the Heuristic

Both the admissible and inadmissible heuristics given in Table 1 *estimate* the cost of reaching a state that satisfies a given predicate. These estimates can be interpreted as random variables with pdf's that describe the distribution of inaccuracy in the estimates. For a given true remaining cost, the heuristic estimate varies over some range of values and according to some density. Functions describing the error density can be estimated by collecting heuristic and actual path costs over a large number of states in a large number of models.

The error density shown in Figure 1 was obtained from the true and estimated costs of 3,759,248 states in 14 different models. The values have been scaled by a linear function of the true remaining path cost, and outliers have been removed. The error density for the inadmissible heuristic follows a roughly normal distribution. The curve overlaid in Figure 1 is the normal distribution with mean and variance taken from the points in the histogram. This is expected because our inadmissible heuristic, shown in Table 1, was constructed to balance overestimation and underestimation. In contrast, the admissible heuristic is severely skewed toward underestimation, since it must always underestimate costs. The admissible heuristic can not be modelled by a normal distribution.

Applying Bayes method to heuristic search requires an estimate of the variance of error density (as will be explained later). For our experiments with BHS, we used the variance computed over all 3,759,248 states–including outliers. The results in Section 5 suggest that the exact value of the observed variance is relatively unimportant to the performance of BHS. Nevertheless, it is important that we have a characterization of how well the heuristic estimates the remaining path length (in this work, a normal distribution with experimentally derived variance). Note also that the true path costs observed in this study of the heuristic error are not used in the meta-heuristic, only the variance of error density. We do *not* use information from this study to bias

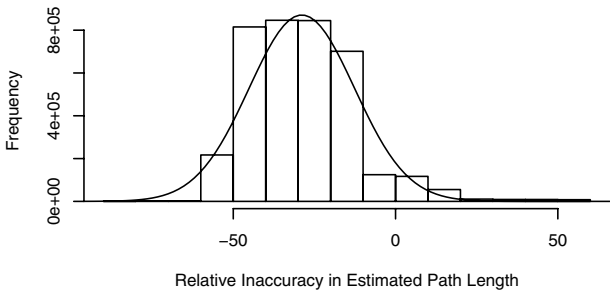


**Figure 1. Observed distribution of error in the inadmissible heuristic and the normal distribution with the same mean and variance as the observed error.**

our algorithm toward a path we already know to be best. In this sense BHS is *not* a statistical learning algorithm.

## 3 Meta-heuristic

Bayes heuristic search differs from conventional heuristic search procedure in that it uses a Bayes meta-heuristic to improve estimates. Before establishing the mathematical properties of the Bayes meta-heuristic, we describe the rationale for the meta-heuristic with an example and then present the algorithm for computing it.

The purpose of the meta-heuristic, and Empirical Bayes method in general, is to improve estimates using information from related sets of estimates. In guided model checking, we claim that estimates for sibling states are more related than estimates for random sets of reachable states. The distribution of heuristic values for a set of sibling states suggests the confidence that should be given to those values. This is illustrated in the following example.

Consider the states and heuristic values shown in Figure 2. States $a$ and $r$ each have five children each. The heuristic estimates for each child are mapped to the first number line. The placement of states on the number line corresponds to their order in the priority queue of states to be expanded. State $a$'s children are clustered between 6 and 9, while state $r$'s children are spread between 5 and 12. Although it is entirely possible that the heuristic values of the children of state $a$ are just as believable as those for the children of $r$, we subjectively argue that siblings with tightly clustered estimates (e.g., states $b$ through $f$) are more plausible than siblings with widely spread estimates (e.g., states $s$ through $w$).

According to this reasoning, the estimates for children of state $a$ are more believable than the estimates for children of state $r$. Some unusual property of either the heuristic or the children of state $r$ must be causing the heuristic to give widely varying and inaccurate values for the children of state $r$. Note that we are not arguing that siblings of a single node are always tightly clustered, only that one would generally expect siblings to have *closer* heuristic values than non-siblings (two arbitrary nodes in the search tree). This assertion is always true for models in which the transition graph allowed all transitions to be reversed. In this case the heuristic estimate for two siblings should never differ by more than two: one transition back to the parent and one transition to a sibling.

A mathematical argument to justify this reasoning is presented in Section 4.

We give the states under $a$ preference using the following algorithm, which can be integrated into any guided search that uses a heuristic to rank states:

- Compute heuristic values, $y_i$, for each child, $i$, of a given state. For example, in Figure 2, states $b$ through $f$ have heuristic values $y_b$ though $y_f$.
- Compute the mean, $\theta_{sib}$, and the variance, $\sigma^2_{sib}$, of the heuristic values of the children.
- For each child $i$: create a revised estimate, $\theta^*_i$, using the weighted average of the original value, $y_i$, and the mean of the siblings, $\theta_{sib}$. For example, if $i = b$ then the revised estimate for state $b$ is

$$\theta^*_b = (1 - B)y_b + B\theta_{sib} \quad (1)$$

  where $B = \sigma^2_y/\sigma^2_{sib}$ and $\sigma^2_y$ is the general variance of the heuristic which was determined above in Section 2.3.

In Figure 2, the values on the second number line show new state ordering resulting from the revised estimates. The low variance of the children of state $a$ implies that their original estimates are more plausible and require little modification (on a per state basis). However, the estimates for state $r$'s children have a higher variance and require more modification.

The significance of this example is, that although state $s$ has a lower estimate than state $b$, the higher variance and mean of state $s$'s siblings lead us to revise the estimate for state $s$ to 8.6 while state $b$'s estimate is revised to 7.4. Hence, the more believable state $b$ will be expanded before state $s$.

## 4 Bayesian Justification for the Meta-heuristic

In this section we map the meta-heuristic described in the preceding section to a Bayesian model and prove the re-

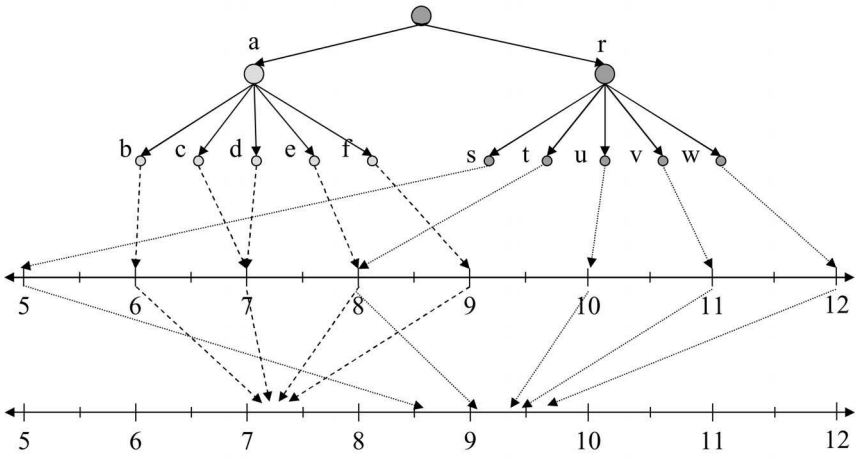


**Figure 2. Original and revised heuristic estimates for two sets of sibling states.**

| Distribution | Description | Notation |
|---|---|---|
| Prior | Characterization of our understanding of some unknown random parameter $\theta$ | $g(\theta)$ |
| Sampling | Relationship between observations drawn from actual distribution of $\theta$ | $f(y\|\theta)$ |
| Posterior | New understanding of $\theta$ based on an observed $y$ and the prior distribution | $\bar{g}(\theta\|y)$ |
| Marginal | Density of probable sampling values, $y$, given the uncertainty in $\theta$ (from the Prior distribution) and the uncertainty inherent in the Sampling distribution | $\bar{f}(y)$ |

**Table 2. Summary of distributions used in Bayes method.**

sulting adjustment lowers the expected mean squared error. The details of this section may be skipped on first reading.

The meta-heuristic presented in the prior section relies on Equation 1 to adjust heuristic values. Equation 1 is called the James-Stein estimator [10] and is used for estimating parameters in a group of parameters with normal distributions. This estimator can best be understood in the context of a Bayesian model.

### 4.1 Bayesian Notation

In order to introduce Empirical Bayes and to motivate our meta-heuristic, we need some basic Bayesian notation. Table 2 summarizes the notation for and definitions of distributions used in this discussion.

Suppose we wish to make a decision, such as which state we should expand first in a transition graph, based on an unknown, random parameter $\theta$ (i.e., the true remaining path length to the goal,). We use a *prior distribution*, or just *prior*, to characterize our belief about the most likely value of $\theta$. For example, see "prior" in Figure 3. The notation $\theta \sim g(\theta)$ indicates that a pdf $g$ models our understanding of $\theta$.

Suppose that we are permitted to observe $y$, an imprecise estimate of $\theta$. Also, suppose that the distribution of the random variable $y$ is a function of $\theta$. In other words, the distribution of $y$ is more related to $\theta$ than a uniform random distribution. We call the distribution of $y$ given $\theta$ the *sampling distribution* and describe it with the notation: $y|\theta \sim f(y|\theta)$ where $f$ is the pdf of $y$ given $\theta$. The sampling distribution describes how far off the estimate $y$ might be from $\theta$ *as if we knew* $\theta$. For example see "sample" in Figure 3.

Bayes law allows us to create a *posterior distribution* which combines our prior understanding of $\theta$ (which is modelled by the pdf $g(\theta)$) with the new understanding of $\theta$ that can be *inferred* by observation of $y$. This yields a new distribution over $\theta$, given the data we have observed:

$$\theta|y \sim \bar{g}(\theta|y) = \frac{f(y|\theta)g(\theta)}{\bar{f}(y)}, \tag{2}$$

where $\bar{g}$ is the pdf of the new distribution on the random variable $\theta|y$. That is, our belief about the parameter $\theta$ given both our prior belief and the observed data $y$ obtained from the heuristic. See the "posterior" in Figure 3.

The density $\bar{f}$ used above, is the *marginal* or *uncondi-*

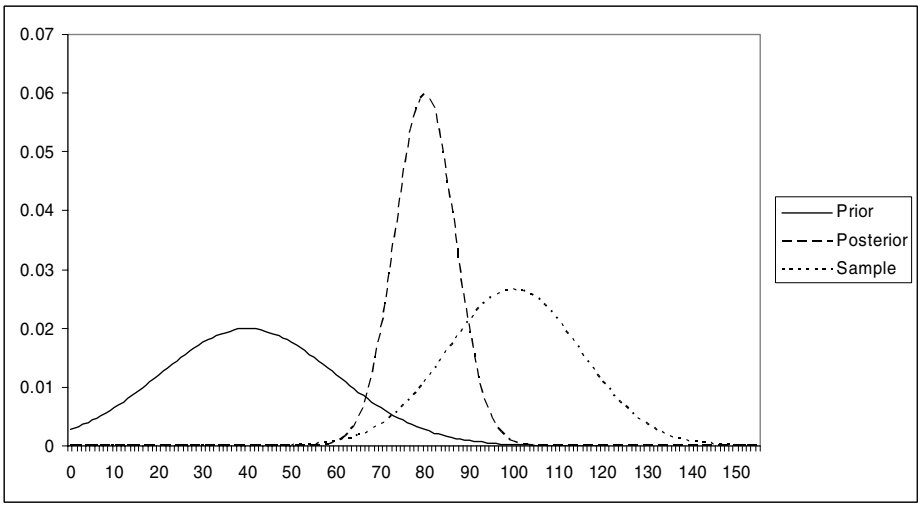


**Figure 3. Prior, posterior and sample distributions in Bayes method.**

*tional distribution* of $y$ and is defined as

$$y \sim \bar{f}(y) = \int_{\Theta} f(y|\theta)g(\theta)d\theta. \quad (3)$$

This distribution models the density of possible values of $y$ given that there is uncertainty both in "$y$ given $\theta$" and in $\theta$–upon which $y$ depends.

In BHS, we model these distributions with the normal distribution in which the prior distribution is

$$\theta \sim Normal(\theta_0, \sigma_0^2) \quad (4)$$

and the sampling distribution is:

$$y|\theta \sim Normal(\theta, \sigma_y^2). \quad (5)$$

In the case of BHS this is the quality of our heuristic, given the actual distance to a goal state. This is the pdf which we estimated empirically in Section 2.

Applying Bayes law to these two distributions yields the posterior distribution:

$$\theta|y \sim Normal((1-B)y + B\theta_0,\ \sigma_y^2(1-B)) \quad (6)$$

where $B = \frac{\sigma_y^2}{\sigma_y^2+\sigma_0^2}$ and the marginal distribution is

$$y \sim Normal(\theta_0, \sigma_y^2 + \sigma_0^2). \quad (7)$$

See [11] for a more detailed development of these relationships.

## 4.2 Empirical Bayes

Thus far, the origin of the prior distribution has not been discussed. The generation of prior distribution when humans are available to render opinions is problematic, but is even more difficult in the context of software because computers do not have prior opinions. To solve this problem, we create a prior opinion for the computer in an algorithmic fashion. The approach is based on an Empirical Bayes philosophy. In Empirical Bayes, we focus on the interpretation and mathematical form of the marginal distribution referenced in Equations 3 and 7. As referenced above, this is the distribution one would observe if a sample of $y$'s were drawn unconditionally on $\theta$, or expressed more algorithmically, first draw a $\theta$ from $g(\theta)$ then draw a $y$ from $f(y|\theta)$. We will refer to this as the Empirical Bayes sample.

In the BHS meta-heuristic, sibling states are treated as the Empirical Bayes sample. Each child state has its own unknown true path length, $\theta_i$, to the target but we observe only the $y_i$s, which are uncertain measures of these path lengths.

Given this set of sibling states with corresponding heuristic values, $y_i$, we estimate the mean ($\theta_{sib}$) and variance ($\sigma_{sib}^2$) of the marginal distribution in Equation 7. From this equation we set $\theta_0$ to $\theta_{sib}$. Likewise, the observed sample variance $\sigma_{sib}^2$ is set equal to the variance equation $\sigma_y^2 + \sigma_0^2$ found in 7. Since $\sigma_y^2$ was derived experimentally in Section 2.3, we solve for the unknown $\sigma_0^2 = \sigma_{sib}^2 - \sigma_y^2$. Given a set of siblings, we have now empirically derived a shared prior distribution specified by $\sigma_0^2$ and $\theta_0$. Using these shared parameters and an individual heuristic value $y_i$, we can compute the parameters of the posterior as shown in Equation 6.

For the purposes of determining the next state to expand we retain only the mean, $\theta_i^*$, of each of these posterior distributions which, from Equation 6, is

$$\theta_i^* = (1-B)y_i + B\theta_0 \quad (8)$$

with $B = \sigma_y^2/\sigma_y^2 + \sigma_0^2$ as before. Re-writing the equation for $B$ using the fact that $\sigma_y^2 + \sigma_0^2 = \sigma_{sib}^2$ we obtain Equation 1, the equation used by the BHS meta-heuristic. Equation 8 is the James-Stein estimator which will have lower mean squared error than using the $y_i$ values alone [10, 12, 13]. More formally, for $n > 2$,

$$\sum_{i=1}^{n} E[(\theta_i^* - \theta_i)^2] <= \sum_{i=1}^{n} E[(y_i - \theta_i)^2]. \qquad (9)$$

We assumed in Section 3 that siblings were more likely to have similar heuristic values than non-siblings. Interestingly, Equation 9 holds even if this assumption does not hold.

Readers familiar with Bayesian methods may prefer a pure Bayesian approach over the "Empirical Bayes" approach used here. Note, however, that we assume all information available to the creator of the heuristic would already be factored into the heuristic, including any information from prior knowledge (using Bayes law in the usual way). The BHS meta-heuristic would be applied *after* all such information has been incorporated into the heuristic. We use an Empirical Bayes model to motivate the structure of our meta-heuristic, and properties of the James-Stein estimator that allow us to achieve heuristics with lower total expected mean squared error.

A more sophisticated Hierarchical Bayes model could also have been used as the basis for the meta-heuristic. The current Empirical Bayes version adds very little overhead to the search procedure since is requires only a few floating point computations per sibling. A sophisticated Hierarchical Bayes model would likely take many more operations and could significantly slow the computation of the meta-heuristic. Such an approach would almost certainly improve our assessment of the posterior variance, but since we discard the posterior variance anyway, it seems unlikely to be worth the computational effort. We plan to consider this issue more in future work.

## 5 Results

We have implemented guided search using the inadmissible heuristic (referred to as just "inadmissible" henceforth) in Table 1 and the Empirical Bayes meta-heuristic based on the inadmissible heuristic (referred to as BHS) in the Murphi model checker [14][1]. This section contains the results of a series of experiments that compare the number of states explored before finding an error using breadth-first search (BFS), inadmissible and BHS.

A comparison with the admissible heuristic is not given because we were unable to define an admissible heuristic sufficiently precise to effectively guide the search. In [9], a comparison of best-first search with an inadmissible heuristic and A* search with an admissible heuristic does not clearly endorse either technique. Our admissible heuristic is less precise than the heuristic in [9] because our heuristic can not exploit properties of queues in SPIN models and must allow multiple variable assignments in an atomic step. Indeed, preliminary experiments with our admissible heuristic required exploring many times more states than best first search with our inadmissible heuristic.

Recall that Equation 1, upon which BHS depends, requires:

1. The mean heuristic value of the sibling nodes in the search ($\theta_{sib}$)
2. The variance of the heuristic values of the sibling nodes ($\sigma_{sib}^2$)
3. The heuristic value to be adjusted ($y$)
4. The variance of the heuristic in general ($\sigma_y^2$)

All of these are easily available in the context of the search except $\sigma_y^2$. We could treat $\sigma_y^2$ as a tuneable parameter, however, Section 2.3 gives us an empirical justification for the value of the parameter. As described there, by exhaustive searching models and comparing the actual path length to the closest error with the heuristic estimate of path length, we can obtain an estimate of $\sigma_y^2$.

7,323,200 was the largest of our empirical estimates of $\sigma_y^2$ which we obtained from the "queens8" model. 3,780,000 was the average variance we observed in the 14 models which we exhaustively searched. All but one of these 14 models had variances above this mean value. We obtained the best results from BHS using the largest variance as shown in the column "BHS (worst $\sigma_y^2$)" in Table 3. Similar results were obtained using any variance from the largest down to the average variance. Results from BHS using the mean variance are shown in the column labelled "BHS (mean $\sigma_y^2$)". Strictly speaking we should exclude the results from the "queens8" model from our results since the estimated variance was obtained from that model. However, given BHS's insensitivity to this parameter, we have included it in the table.

The Bayesian model which supports BHS assumes normally distributed errors. In practice BHS does not seem sensitive to this assumption. In simulation studies, using uniformly distributed error, the performance of BHS fell only 11% [15].

In 15 of 22 of the problems in Table 3 , the inadmissible search located an error after exploring fewer states than BFS. The BHS located an error after exploring the same or fewer states than the inadmissible search in 17 of 22 problems. In 7 cases, the BHS explores between one and two orders of magnitude fewer states than the inadmissible search.

[1] The model checker and all models used in this paper are available at http://vv.cs.byu.edu

| Problem | BFS | Inad-missible | BHS (worst $\sigma_y^2$) | BHS (mean $\sigma_y^2$) | Inadmiss-able mean squared error | BHS mean squared error |
|---|---|---|---|---|---|---|
| atomix | 1,573,909 | 484,484 | 12,493 | 12,698 | 99,791,000 | 99,790,000 |
| atomix1 | 1,573,909 | 225,226 | 12,419 | 10,846 | 99,791,000 | 99,747,000 |
| atomix-2e | 39,321,000* | 12,147,404 | 368,269 | 326,320 | 99,806,000 | 99,450,000 |
| atomix3 | 17,476,000* | 446,783 | 25,713 | 11,125 | 99,960,004* | 99,950,645.05* |
| atomix4 | 22,130,000* | 2,288,795 | 1,640,343 | 1,471,997 | 99,960,004* | 99,948,005.54* |
| adash1212e | 3,742 | 4,065 | 6,296 | 8,305 | 91,859,000 | 99,797,000 |
| adash1312e | 84,775 | 95,926 | 342,000* | 257,000* | 9,915,900 | 71,960,000 |
| 2-peterson | 70 | 70 | 20 | 20 | 32,794,000 | 29,842,400 |
| 12-peterson | 12,141,000* | 4,677,000* | 4,321 | 4,310 | 99,500,625* | 99,489,611.77* |
| lin | 173 | 101 | 101 | 101 | 99,902,989.65 | * |
| dense | 1,380,098 | 1,533,833 | 193 | 201 | 5,071.73 | 4,575.97 |
| down | 10,957 | 10,957 | 517 | 517 | 99,779,700 | 99,769,623.81 |
| bulls&cows | 7,354 | 96 | 96 | 96 | 99,828,100 | 99,822,057.06 |
| arbiter5 | 4,540 | 5,279 | 1,983 | 3,019 | 996,386,00 | 211.79 |
| queens8 | 3,696,597 | 190,390 | 65 | 65 | 99,996,000 | 99,728,000 |
| sets | 29 | 18 | 18 | 18 | 664,884 | 634,454.36 |
| sort5 | 4 | 6 | 6 | 6 | 6,547,710 | 6,498,550 |
| ss2 | 221 | 40 | 438 | 92 | 98,423,700 | 98,376,900 |
| td | 2,768,951 | 247 | 247 | 247 | 98,629,024,089 | 99,164,303.92 |
| scierr | 64 | 62 | 75 | 75 | 99,960,004 | 99,954,916.27 |
| mutex2 | 23,863 | 23,862 | 7,964 | 7,965 | 1,336,758.67 | 1,226,216.94 |
| mutex3 | 105,400 | 15,395 | 55,687 | 55,683 | 47,822,128.69 | 45,319,758.89 |

**Table 3. States explored before error discovery in breadth first search, guided search using the inadmissible heuristic and BHS with worst and mean variance. Mean squared error for inadmissible heuristic and BHS. An * indicates that the search exhausted 2 GB of memory, and failed, before locating an error or completing the mean squared error computation.**

Reducing the mean squared error, even by a small amount, is correlated with reducing the number of states explored. In 15 of 18 problems for which the BHS meta-heuristic reduced the mean squared error, the BHS also reduced the number of states explored.

The BHS is significantly less efficient than both the BFS and the inadmissible searches on the adash1x12e models. The BHS also has a higher mean squared error than inadmissible search on those models. The adash1x12e models are abstract models of the Stanford DASH processor used in [16]. The poor performance of the BHS may be caused by nested quantifiers over enumerated types. The property to be verified for adash1x12e includes eleven quantifiers over enumerated types nested to a depth of five. The other models include at most two non-nested quantifiers. Since the heuristic function gives meaning to the magnitude of a value in a state (e.g., the difference between the goal value and the current value in an equality) and values are irrelevant in enumerated types, the heuristic makes poor estimates for the adash1x12e models. The BHS meta-heuristic may simply amplify poor estimates.

Figure 4 is obtained from a set of 300 experiments in which the BFS, inadmissible search and BHS algorithms locate winning states, encoded as violations of a safety property, in 100 instances of atomix. This experiment measures how many states must be explored before error discovery in a given percentage of models using each search technique. Atomix is a single player game for which finding solutions is PSPACE-complete [17, 18]. In atomix, a player moves atoms and must use them to construct a target molecule. The playing field contains barriers as well as atoms. When an atom is moved in any direction it continues to move until it reaches the border of the board, a barrier or another atom. The player wins when the molecule is correctly assembled. The one hundred variations were created using a four by four board containing one barrier (more barriers reduce reachable states and search times). The target molecule contained five atoms that must be arranged in the correct positions. The models were generated with random placement of both the atoms and barrier. There are 5,765,760 such models. Models that did not contain a winning state were excluded.

Figure 4 graphs the number of states explored against the number of models in which an error had been discovered after exploring the given number states or fewer. For example, BHS discovered errors in 75 models after exploring 75,000 or fewer states in each model. BFS discovers an error in all one hundred models only after exploring 3,541,600 or fewer states per model (not shown in Figure 4). These results suggest that performance of BHS is insensitive to the problem instance.

## 6 Conclusion

Interpreting heuristic estimates as random variables rather than point values allows the application of statistical methods to improve the expected behavior of guided search. More specifically, the BHS algorithm reduces the expected mean squared error of a given heuristic function. The reduction appears insensitive to the variance of error in the estimated path length, the distribution used to model error and the specific problem instance. In our experiments, reducing the mean squared error reduces the number of states explored before finding an error by as much as four orders of magnitude.

Avenues for future work include applying the meta-heuristic to structural heuristics in addition to property heuristics. We expect similar positive results since the reduction in mean squared error obtained by the meta-heuristic does not depend on the structure of the heuristic. The meta-heuristic can be incorporated into our parallel algorithm for concentrating search effort on regions likely to contain errors in large transition graphs [19]. This parallel algorithm currently uses random walk to find regions likely to contain errors. While random walk is surprisingly effective in some cases, replacing random walk with guided search may improve the coverage in other cases.

## References


[1] P. Hart, N. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems, Science and Cybernetics*, vol. SSC-4, no. 2, pp. 100–107, 1968.

[2] A. M. O. Hansson, "Probabilistic heuristic estimates," *Annals of Mathematics and Artificial Intelligence*, vol. 2, pp. 209–220, 1990.

[3] T. A. L. Bradley P. Carlin, *Bayes and Empirical Bayes Methods for Data Analysis*. Chapman & Hall, 1996.

[4] J. Maritz, *Emperical Bayes Methods*. Methuen, 1970.

[5] C. Yang and D. Dill, "Validation with guided search fo the state space," in *35th Design Automation Conference (DAC98)*, pp. 599–604, 1998.

[6] R. Bloem, K. Ravi, and F. Somenzi, "Symbolic guided search for CTL model checking," in *Design Automation Conference (DAC'00)*, pp. 29–34, 2000.

[7] A. Groce and W. Visser, "Model checking Java programs using structural heuristics," in *International Symposium on Software Testing and Analysis*, pp. 12–21, July 2002.

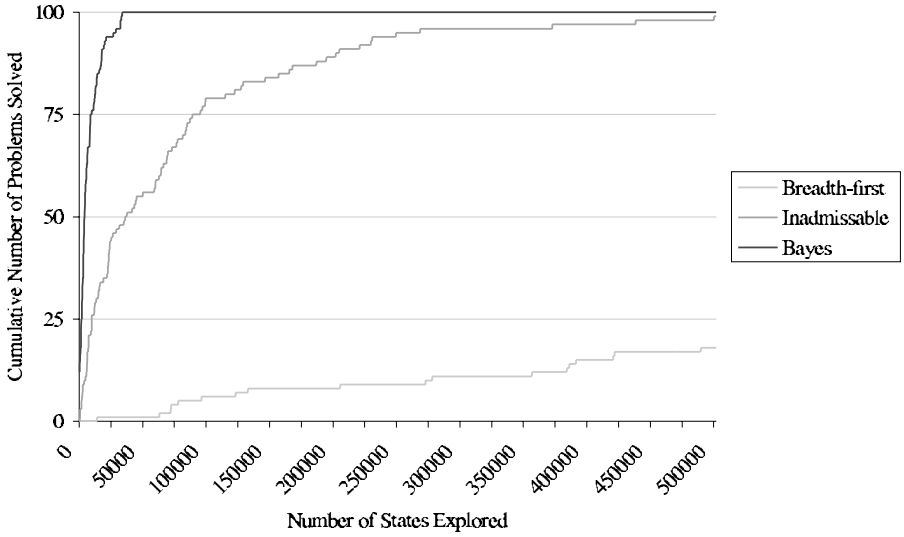


**Figure 4. States explored before error discovery in one hundred mutations of atomix.**


[8] T. Pyhälä and K. Heljanko, "Specification coverage aided test selection," in *Proceeding of the 3rd International Conference on Application of Concurrency to System Design (ACSD'2003)* (J. Lilius, F. Balarin, and R. J. Machado, eds.), pp. 187–195, IEEE Computer Society, June 2003.

[9] S. Edelkamp, A. Lluch-Lafuente, and S. Leue, "Directed explicit model checking with HSF-SPIN," in *8th International SPIN Workshop on Model Checking Software*, no. 2057 in Lecture Notes in Computer Science, Springer-Verlag, 2001.

[10] C. Stein, "Inadmissibility of the usual estimator for the mean of a multivariate normal distribution," in *Proceedings of the Third Berkeley Symposium on Mathematical Statistics and Probability 1*, pp. 197–206, University of California Press, 1955.

[11] M. H. DeGroot, *Optimal Statistical Decisions*. McGraw-Hill, 1970.

[12] C. N. Morris and B. Efron, "Empirical Bayes estimators on vector observations - an extension of Stein's method," *Biometrika*, vol. 59, pp. 335–347, 1972.

[13] C. N. Morris, "Parametric empirical Bayes inference: theory and applications. with discussion," *Journal of the American Statistical Association*, vol. 78, no. 381, pp. 47–65, 1983.

[14] D. L. Dill, "The Mur$\phi$ verification system," in *Computer-Aided Verification, CAV '96* (R. Alur and T. A. Henzinger, eds.), vol. 1102 of *Lecture Notes in Computer Science*, (New Brunswick, NJ), pp. 390–393, Springer-Verlag, July/Aug. 1996.

[15] K. Seppi, "Empirical Bayes assisted probabilistic search," Tech. Rep. VV-0301, Brigham Young University Department of Computer Science, 2003.

[16] U. Stern and D. L. Dill, "A new scheme for memory-efficient probabilistic verification," in *IFIP TC6/WG6.1 Joint International Conference on Formal Description Techniques for Distributed Systems and Communication Protocols, and Protocol Specification, Testing, and Verification*, 1996.

[17] F. Hueffner, S. Edelkamp, H. Fernau, and R. Niedermeier, "Finding optimal solutions to atomix," in *Proceedings of the Joint German/Austrian Conference on Artificial Intelligence, 24th German / 9th Austrian Conference on Artificial Intelligence (KI 2001)*, no. 2174 in Lecture Notes in Artificial Intelligence, pp. 229–243, Springer, 2001.

[18] M. Holzer and S. Schwoon, "Assembling molecules in atomix is hard," Tech. Rep. TUM-I0101, Institut für Informatik, Technische Universität München, June 2001.

[19] M. D. Jones and J. Sorber, "Parallel search for LTL violations," *Software tools for technology transfer*, 2004. To appear.