# REPORT DOCUMENTATION PAGE

*Form Approved*
*OMB No. 0704-0188*

| | | |
|---|---|---|
| **1. REPORT DATE** *(DD-MM-YYYY)*<br>01-17-2012 | **2. REPORT TYPE**<br>Final | **3. DATES COVERED** *(From - To)*<br>4/1/08 – 11/30/10 |

| | |
|---|---|
| **4. TITLE AND SUBTITLE**<br>Bridging the Gap Between Theory and Practice: Structure and Randomization in Large Scale Combinatorial Search | **5a. CONTRACT NUMBER** |
| | **5b. GRANT NUMBER**<br>FA9550-08-1-0196 |
| | **5c. PROGRAM ELEMENT NUMBER** |
| **6. AUTHOR(S)**<br>Carla P. Gomes | **5d. PROJECT NUMBER** |
| | **5e. TASK NUMBER** |
| | **5f. WORK UNIT NUMBER** |
| **7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)**<br>Cornell University<br>Office of Sponsored Programs<br>373 Pine Tree Rd.<br>Ithaca, NY 14850-2820 | **8. PERFORMING ORGANIZATION REPORT NUMBER** |
| **9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)**<br>Air Force Office of Scientific Research<br>875 N. Randolph St. Room 3112<br>Arlington, VA 22203 | **10. SPONSOR/MONITOR'S ACRONYM(S)** |
| | **11. SPONSOR/MONITOR'S REPORT NUMBER(S)**<br>AFRL-OSR-VA-TR-2012-0868 |

**12. DISTRIBUTION / AVAILABILITY STATEMENT**

Approved for public release

**13. SUPPLEMENTARY NOTES**

**14. ABSTRACT**

This research effort focused on three core research challenges: (1) How to explain the gap between formal analysis and practical performance for combinatorial search; (2) How to characterize and capture hidden tractable structure in real-world problems; and, (3) How to further boost combinatorial search methods for real-world problems. A series of advanced formal models for predicting the runtime of combinatorial search methods were developed. Models of runtime distributions of search methods capturing exponential and power law (heavy-tailed) regimes for both complete and incomplete randomized search methods were introduced, together with a generative model that generates search trees with any pre-defined degree of heavy-tailedness. New methods for the efficient computation of the number solution clusters and their marginal distributions were developed. The notion of "backdoor sets," – a measure that characterizes hidden problem structure – was extended to encompass combinatorial optimization problems as well as learning during search, thereby providing novel insights into the connection between the hidden structure of optimization problems and the surprising efficiency of today's optimization engines. A novel Markov Chain Monte Carlo sampling strategy, inspired by a flat histogram method from statistical physics, was developed to compute the density of states of a Boolean formula. Multi-agent inference problems in dynamic environments were formulated into the framework of message passing algorithms and graphical models, generalizing the standard Kalman filter to the distributed case. A new hybrid strategy for the MaxSAT problem was also proposed, combining the complementary strength of local search and systematic search, bringing the best of both worlds in a way that is ideal for current multi-core architectures.

**15. SUBJECT TERMS**

| | | | | | |
|---|---|---|---|---|---|
| **16. SECURITY CLASSIFICATION OF:** | | | **17. LIMITATION OF ABSTRACT** | **18. NUMBER OF PAGES** | **19a. NAME OF RESPONSIBLE PERSON** |
| **a. REPORT**<br>U | **b. ABSTRACT**<br>U | **c. THIS PAGE**<br>U | | | **19b. TELEPHONE NUMBER** *(include area code)* |

**Standard Form 298 (Rev. 8-98)**
**Prescribed by ANSI Std. Z39.18**

# Bridging the Gap Between Theory and Practice: Structure and Randomization in Large Scale Combinatorial Search

FA9550-08-1-0196

Grant Period:

4/1/08 – 11/30/10

PI: Carla P. Gomes
gomes@cs.cornell.edu
Department of Computer Science

Cornell University

Final Report

## Abstract

This research effort focused on three core research challenges: (1) How to explain the gap between formal analysis and practical performance for combinatorial search; (2) How to characterize and capture hidden tractable structure in real-world problems; and, (3) How to further boost combinatorial search methods for real-world problems. Predicting the runtime of combinatorial search methods is a notoriously hard problem due to tremendous variations in runtime observed when solving practical problem instances. A series of advanced formal models for predicting the runtime of combinatorial search methods was developed. Models of runtime distributions of search methods capturing exponential and power law (heavy-tailed) regimes for both complete and incomplete randomized search methods were introduced, together with a generative model that generates search trees with any pre-defined degree of heavy-tailedness. In order to better understand and model solution spaces of combinatorial problems, new methods for the efficient computation of the number solution clusters and their marginal distributions were developed. These methods can effectively handle practical problem instances with tens of thousands of variables, containing solution clusters with sizes ranging over many orders of magnitude. Reasoning based on such clusters has been the key component of highly successful combinatorial search methods proposed recently. The notion of "backdoor sets," --- a measure that characterizes hidden problem structure --- was extended to encompass combinatorial optimization problems as well as learning during search, thereby providing novel insights into the connection between hidden structure of optimization problems and the surprising efficiency of today's optimization engines. Probabilistic reasoning techniques based on message passing, namely belief propagation and survey propagation, were analyzed in the context of combinatorial problems in the Boolean satisfiability domain, resulting in the first detailed study of the evolution of these search methods over time as well as the utilization of these techniques to provide statistical estimates on key properties of the solution space. The problem of computing the density of states of a Boolean formula, which is a generalization of Satisfiability Testing, MAX-SAT, and model counting, was also studied and a novel Markov Chain Monte Carlo sampling strategy, inspired by a flat histogram method from statistical physics, was developed. The new sampling method provides novel insights into combinatorial search spaces that lie far beyond the reach of previous techniques. Multi-agent inference problems in dynamic environments were formulated into the framework of message passing algorithms and graphical models, generalizing to the distributed case of the Kalman filter. A new hybrid strategy for optimizing the MaxSAT problem was proposed, combining the complementary strength of local search and systematic search, bringing the best of both worlds in a way that is ideal for current multi-core architectures.

## 1. Introduction

In the last decade we have witnessed tremendous progress in the design and development of search algorithms for solving combinatorial problems. For example, consider progress in the complete or exact backtrack-style methods for constraint satisfaction problems (CSPs), and in particular Boolean satisfiability (SAT) problems. In the early 1990s we could only solve formulas with around 100 variables and 1,000 clauses, whereas current state-of-the-art complete Davis-Putnam- Logemann-Loveland (DPLL) based SAT solvers can now handle much larger real-world instances, with over 1,000,000 variables and over 5,000,000 constraints. We have witnessed similar progress in the area of Integer Programming. Current complete state-of-the-art solvers for combinatorial problems seem to defy the theoretical worst-case results for solving real-world instances of hard computational problems.

The research covered under this grant focused on three key research questions: (1) How to explain the performance gap between theory and practice for combinatorial problems. (2) How to characterize and capture hidden tractable structure in real-world problems. (3) How to further boost combinatorial search methods for real-world problems.

Our work brings together techniques from constraint programming, mathematical programming, and satisfiability in a symbiotic way to address the three research questions. In order to evaluate the different approaches and methods, we considered a range of real-world benchmark problems from hardware and software verification to the design of experimental experiments, as well as more abstract problem domains such as combinatorial design, random constraint satisfaction problems (CSP), and random satisfiability (SAT).

In the next sections we highlight our research accomplishments during the period of the grant which relate to the research questions and themes identified above. In Section 2 we describe our research on randomized search procedures and runtime distributions of search procedures. Section 3 describes methods for counting and sampling solutions of combinatorial problems. Section 4 studies the problem of uncovering so-called hidden structure in combinatorial problems. Section 5 describes methods for over-constrained problems and Section 6 presents methods for inference for dynamic processes.

## 2. Randomization and runtime distributions of search methods

Several factors have contributed to the tremendous progress that we have observed in the design and development of new algorithmic techniques and solvers for combinatorial problems, in addition to the increase in computational power. In particular, these factors include more sophisticated data-structures, non-chronological backtracking, fast pruning and propagation methods, nogood (or clause) learning, combination of branching and cuts, and more recently randomization and restarts. See e.g., [9], for background literature.

Randomization has greatly extended our ability to solve hard computational problems. In general, however, we think of randomization in the context of local search. While local search methods have proven to be very powerful, in some situations they cannot supplant complete or exact methods due to their inherent limitation: local search methods cannot prove inconsistency or optimality. Surprisingly, randomization and restarts have also been shown quite effective for complete backtrack-style search methods. In fact, randomization and restarts are now an integral part of most state-of-the-art complete SAT and CSP solvers.

The discovery of the effectiveness of randomization and restart strategies in complete or exact search methods was made in the context of the study of the runtime distributions of backtrack-style algorithms. For a long time researchers had observed that the performance of backtrack-style search methods can vary dramatically depending on the way one selects the next variable to branch on (the "variable selection heuristic") and on what order the possible values are assigned to a variable (the "value selection heuristic"). In fact, quite often the branching heuristics provide incorrect search guidance, forcing the procedure to explore large sub-trees of the search space that do not contain any solution. As a consequence, backtrack-search methods exhibit a large variance in performance. For example, we see significant differences on runs of different heuristics, runs on different problem instances, and, for randomized backtrack-search methods, significant differences on runs with different random seeds. The inherent exponential nature of the search process appears to magnify the un-predictability of search procedures. In fact, it is not uncommon to observe a backtrack-search procedure "hang" on a given instance, whereas a different heuristic, or even just another randomized run, solves the instance quickly.

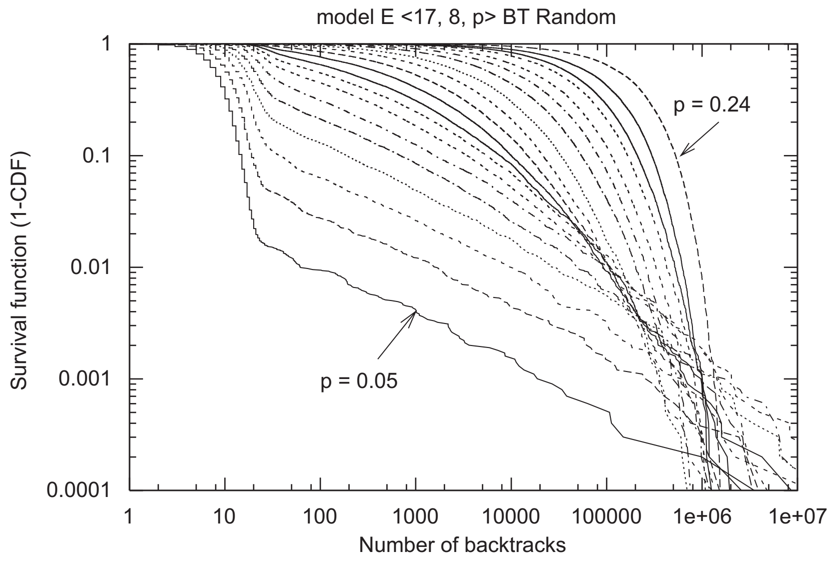


**Figure 1 The progression from heavy-tailed regime to non-heavy-tailed regime. Log–log plot of the survival functions of the runtime distributions of a backtrack-search algorithm on constraint satisfaction instances of model E (17, 8, p), for different values of p [9].**

Although researchers had been well aware of the high variance of backtrack-search algorithms, the discovery of the so-called heavy-tailed nature of the runtime distributions of backtrack-search methods was somehow surprising and even counter-intuitive. Heavy-tailed distributions exhibit power-law decay of the tails. That is why heavy-tailed distributions are also referred to as power-laws. (See Figure 1). The power-law decay of the tail causes it to be heavy, and therefore some of the moments do not converge — heavy-tailed distributions are therefore characterized by infinite moments, e.g., they can have infinite mean, or infinite variance, etc. This is in contrast with non-heavy-tailed distributions characterized by exponential decay. Related to heavy-tailedness is fat-tailedness. The notion of fat-tailedness may be introduced using the concept of kurtosis, and comparing the kurtosis of a given distribution with the kurtosis of the standard normal distribution. The kurtosis of the standard normal distribution is 3. A distribution with a kurtosis larger than 3 is fat-tailed or leptokurtic. Like heavy-tailed distributions, fat-tailed distributions have long tails, with a considerable mass of probability concentrated in the tails. Nevertheless, the tails of fat-tailed distributions are lighter than heavy-tailed distributions. Therefore, contrarily to heavy-tailed distributions, all the moments of fat-tailed distributions are finite. Examples of distributions that are characterized by fat-tails are the exponential distribution and the lognormal distribution. Interestingly, in the context of search, heavy-tails have been observed not only in aggregated runtime distributions of backtrack-search methods, when considering a collection of instances of the same class (e.g., random binary CSP instances generated with the same parameter space), but also when running a randomized backtrack-search procedure on the same instance several times, in which the randomization is only used to break ties in the variable and/or value selection.

The understanding of the fat-tailed and heavy-tailed nature of the distributions underlying backtrack-search methods has led to the design of new search strategies, in particular restart strategies for complete backtrack-search methods. For example, we have shown how randomized restarts of search procedures can dramatically reduce the variance in the search behavior. In fact, we demonstrated that a search strategy with restarts provably eliminates heavy tails. Interestingly, Beame et al. showed that clause learning combined with restarts, as used by current-state-of-the-art SAT solvers, corresponds to a proof system exponentially more powerful than that of DPLL.

While heavy-tailed behavior has been observed in backtrack-search methods, it is clear that it does not occur in all problem instances. In fact, backtrack-style algorithms exhibit dramatically different statistical regimes across the different constrainedness regions of random CSP models—a heavy-tailed regime in the under-constrained area is replaced by a non-heavy-tail regime as one moves towards the phase transition.

### 2.1 A generative power-law search tree model for complete or exact methods

(See [9] for a detailed description of this work and background literature.)

A deep understanding of heavy-tailed phenomena involves formal generative models. In fact, the search for good generative models for power-law distributions is a new active research area across different domains. For example, the so-called model of preferential attachment that generates power-law degree distributions for random graphs is an abstraction for modeling how social networks or the Internet lead to heavy-tailed behavior.

The generation of power-law distributions for backtrack search is also quite challenging, especially if one attempts to capture the full behavior of backtrack search. A compromise is to produce more abstract models, such as the model proposed by Chen et al. In such a model, only high level branching decisions leading to "subtrees of the search space" are modeled. Branching decisions within a given "subtree" are not modeled. Despite its level of abstraction, the model provided interesting insights into search algorithms. For example, it led to the so-called notion of backdoor set, a set of critical variables that captures the combinatorics of the problem with respect to the propagation procedure of the solver: once values are assigned to the backdoor set, the remaining problem is solved by propagation (see also section 3).

Our research contribution to this topic during this grant period was twofold. We showed how the different regimes observed in backtrack-search methods across different constrainedness regions of random CSP models can be captured by a mixture of the so-called stable distributions. Stable distributions capture a range of heavy-tailed and non-heavy-tailed distributions. We also developed a generative search tree model whose distribution of the number of nodes visited during search is formally heavy-tailed. Even though our model is an abstraction of backtrack search, it is more realistic than previous models. In particular, while the model by Chen et al. only considers high level branching decisions leading to "subtrees of the search space", more specifically, subtrees of size $2^0$ , $2^1$ , $2^2$ , . . . , $2^n$ nodes, our model considers finer grained branching decisions, at every node. Furthermore, it allows us to generate search trees with any degree of heavy-tailedness. Our model also captures a key aspect of heavy-tailed behavior in backtrack search—the longer the run the more unlikely it is for the search procedure to stop. This overall behavior is achieved by the fact that the probability of going down the search tree decreases exponentially, combined with the fact that, as one goes down the search tree, the probability of making a "wrong decision" – i.e., not picking a terminal node that corresponds to a solution or that leads to a proof of unsatisfiability – given all the "wrong decisions" so far, increases. These two opposite factors – an overall exponential decrease in going down the search tree and an exponential increase in search space as we go down the search tree – are key to the generation of power-law decay. Our model captures binary trees as well as other tree shapes that more closely resemble the search trees produced in combinatorial search. We should also point out that the nodes in our model capture different decision points, such as picking the next variable to branch on, or picking a value to assign to a variable, or picking a backtracking point, or more generally picking or not picking the "right terminal node". Therefore, our model can be viewed as an abstraction for different variants of backtrack-search models (See Figure 2).

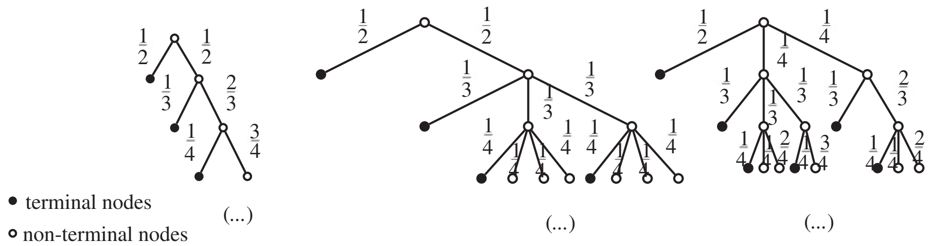


**Figure 2 Different variants of search trees with power law decay (alpha=1)**

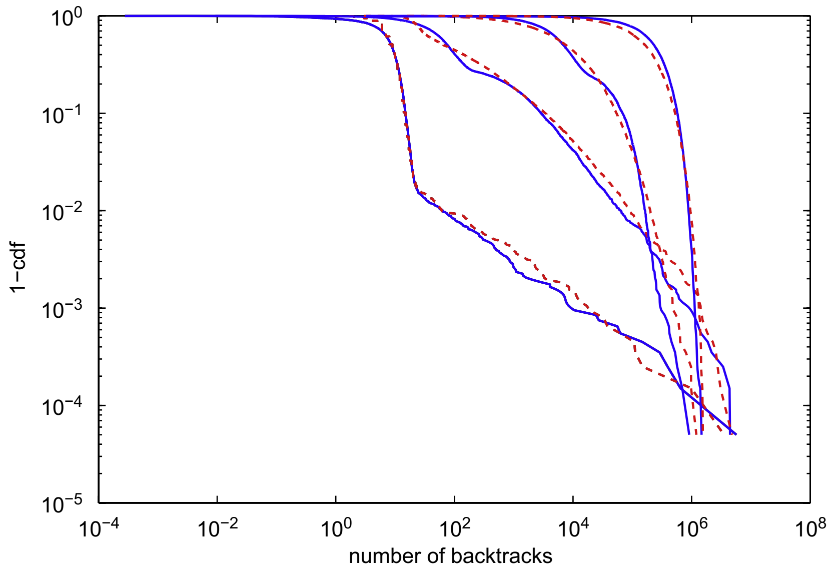


**Figure 3 – Log–log plot of survival functions of stable mixtures (solid line) and empirical runtime distributions of a backtrack-search algorithm on instances of Model E, p = 0.05, 0.11, and 0.24 (dashed line).**

We also showed how the different statistical regimes observed in the runtime distributions of backtrack-style algorithms on random CSP instances can be captured by a mixture of stable distributions, in which one of the components is heavy-tailed and the other component is the normal distribution (See Figure 3). This mixture provides interesting insights: despite the relative high weight of the normal distribution across the different regions, the extremely low alpha values of the heavy-tailed stable produce a heavy-tailed regime; as alpha is increased, the heavy-tailed component no longer outweighs the normal component, leading to exponentially decaying tails. From an algorithmic point of view the heavy-tailed regime corresponds to having an algorithm that has good chances of finding solutions with short runs, given the fact that this regime is in the under- or medium-constrained area. However, now and then it makes a sequence of mistakes that leads to extremely long tails, therefore heavy-tails. As the instances become harder, the heavy-tailed regime is replaced by a non-heavy-tailed regime in which the normal distribution dominates, with a corresponding increase in alpha. In this region the instances become inherently harder, all the runs become homogeneously long, and the algorithm does not have a chance of producing short runs. Therefore, there is a dramatic decrease in the ranges of the runtime distributions and the fast drop of the tails.

In summary, in this work we introduced a generative search tree model that captures key aspects of heavy-tailed behavior in combinatorial search. Furthermore, our model allows us to generate search trees with any degree of heavy-tailedness. We also showed how a mixture of stable distributions captures the statistical regimes observed in runtime distributions of backtrack-style algorithms across different constrainedness regions of random CSP instances. We hope our models will provide further insights into the design of new algorithmic strategies and lead to further improvements in the design of search methods.

### 2.2 Optimal Noise and Runtime Distributions in Local Search

(See [5] for a detailed description of this work and background literature.)

Designing, understanding, and improving, local search methods for constraint reasoning, and in particular for Boolean satisfiability (SAT), has been the focus of hundreds of research papers since the 1990s and even of earlier papers. For SAT, techniques such as greedy local search, tabu search, solution guided search, focused random walk, and reactive or adaptive search have led to much success. Specifically, Walksat stands out as one of the initial solvers that introduced many of the key ideas in use today and, is still competitive with the state of the art.

Many attempts have been made to understand the behavior of local search methods in terms of local minima, exploring "plateaus", the exploration vs. exploitation tradeoff, etc. However our formal understanding is limited mostly to relatively simple variants of local search, such as a pure greedy search, a pure random walk, or a combination of the two. This is not surprising as the techniques employed by Walksat and other state-of-the-art local search solvers are too complex to allow a formal analysis in terms of, for example, a traditional Markov Chain. At the same time, there is a wealth of information available from observations of the behavior of local search methods on a variety of domains, most notably for random 3-SAT. There is either formal or anecdotal evidence of various features, such as Walksat, scaling linearly at optimal noise but exponentially at sub-optimal noise, and there are suggestions that the runtime distribution of local search on a single random instance has an exponentially decaying tail. Our work provides convincing empirical evidence in favor of, or even against, such anecdotal insights and observations. We studied the behavior of Walksat on hard, large, random 3-CNF formulas and investigated its time complexity in relation to the clause-to-variable ratio $\alpha$ and the (static) noise level – both of which Walksat is highly sensitive to. Unlike previous studies, our conclusions are based on very large formulas and are thus free of "small N effects". This might explain the difference between our conclusions and those of, e.g., Hoos and Stutzle.

While many new local search SAT solvers are based on "adaptive" or "dynamic" noise, these solvers are apparently unable to settle on the optimal noise setting for hard random 3-CNF formulas, doing much worse than optimal static noise. E.g., we found that the SAT Competition 2009 winners in the satisfiable Random category, TNM and gnovelty+2, were slower than Walksat at optimal noise by a factor of roughly 4x for N=10,000 variable formulas with $\alpha = 4.2$, 13x for N=20,000, 31x for N=30,000, 54x for N=40,000, and 785x for N=50,000. This also shows that, unlike Walksat, these adaptive noise solvers scale super-linearly in this domain, justifying the interest in our study of static noise.

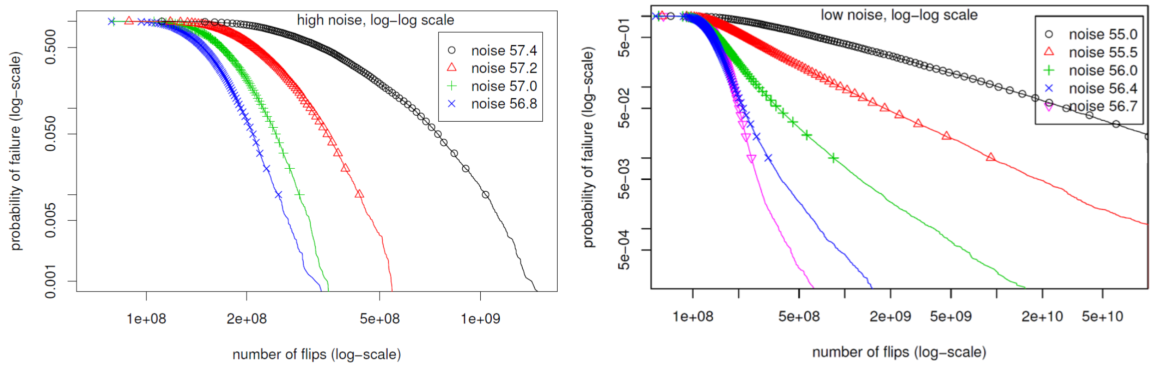


**Figure 4 Log–log plot of survival functions of runtime (number of flips) distributions of the Walksat procedure on a large SAT instance for different noise levels. (Left panel) Non-heavy tailed regime for high noise level. (Rigth panel) Heavy-tailed regime for low noise level.**

Our work showed a surprisingly simple step-linear analytical fit for the value of the optimal noise as a function of $\alpha$, and an equally simple analytical expression for the mean running time of Walksat (measured as the number of flips) at this optimal noise. This fit as well as our data exhibit linear scaling with N for $\alpha$ close to the phase transition region for 3-SAT. Second, we studied the runtime distribution of Walksat on single instances and found the first clear evidence of power-law decay in the probability of failure in T flips in the tail of the distribution. Power-law decays and heavy-tailed runtime distributions have been one of the key observations for DPLL-style systematic search solvers and have led to methodologies such as rapid restarts and algorithm portfolios. This phenomenon, however, is usually not associated with local search. We showed that after a (relatively long) "flat" region, the probability of failure decays exponentially in the high noise regime but as a power-law in the low noise regime. Third, we showed that as Walksat proceeds, the number of unsatisfied clauses exhibits an interesting gradual decay that happens only at near-optimal noise. The kind of empirical study pursued here requires a significant computational power (e.g., 100,000 runs for some low noise levels to observe a clear trend). We used Yahoo!'s Apache Hadoop based M45 cloud computing platform with the net computational effort being equivalent to around 14 years of single CPU time.

### 2.3 Markov Chain Model Capturing Exponential and Power-Law Decay

(See [5] for a detailed description of this work and background literature.)

We developed a preliminary Markov Chain model capturing, e.g., exponential scaling with N and power-law decay at low noise. A model that captures such features and is yet simple to describe and simulate can be a very useful tool for understanding and exploiting the tradeoffs inherent in local search.

The model has two parts. The first part is a linear MC with states corresponding to truth assignments that satisfy the same fraction of clauses of a formula F, with the leftmost state encapsulating all solutions. Second, hanging from each state in the top chain is a "trap gadget", which captures the behavior of Walksat when it "gets lost" exploring parts of the search space without any solutions, leading to the heavy-tail.

## 3. Counting and problem structure

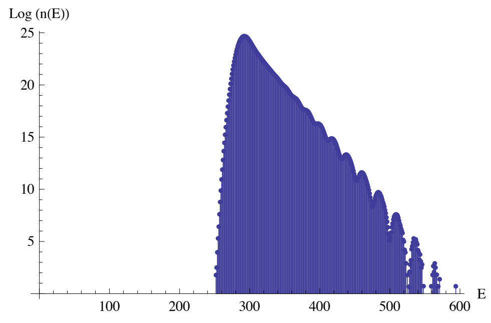
### 3.1 Computing the density of states of a Boolean formula

(See [1 and 3 ] for a detailed description of this work and background literature.)

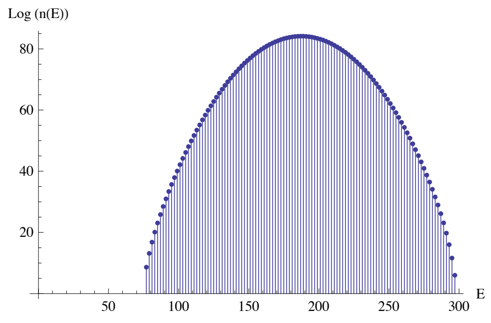
As mentioned above, Boolean satisfiability (SAT) solvers have been successfully applied to a wide range of problems, ranging from automated planning to hardware and software verification. In all these applications, the original problem is encoded into a Boolean formula and the task is that of deciding whether it is satisfiable or not.

Given the tremendous success of SAT solvers, a lot of attention has been directed toward extending this technology to the model counting problem, that is the problem of computing the number of distinct satisfying assignments for a given propositional formula. This task is also very important because of its wide range of applications. For example, several probabilistic inference problems in graphical models such as Bayesian inference can be effectively translated into model counting. Another very active line of research is devoted to the study of the optimization version of SAT, namely the maximum satisfiability problem (MAX-SAT), where the goal is to find a truth assignment that satisfies the maximum possible number of constraints. MAX-SAT is important because it can be effectively used to solve many fundamental graph theoretic problems such as MAX-CUT, MAX-CLIQUE, and Minimum Vertex Cover, and because it has direct applications in a wide range of domains such as routing problems and expert-systems.

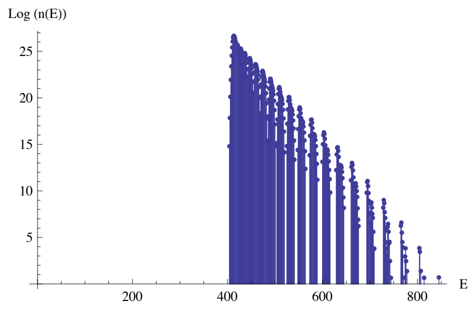
In our work we considered the problem of computing the density of states of a Boolean formula, which is a generalization of Satisfiability Testing, MAX-SAT and model counting. Consider a combinatorial state space S, such as the set of all possible truth assignments to N Boolean variables. Given a partition of S into subsets, we considered the problem of estimating the size of all the subsets in the partition. This problem is also known as computing the density of states. For instance, given a Boolean formula with m constraints, we can partition the set of all possible truth assignments according to the number of constraints they violate. In this case, the density of states gives the size of all



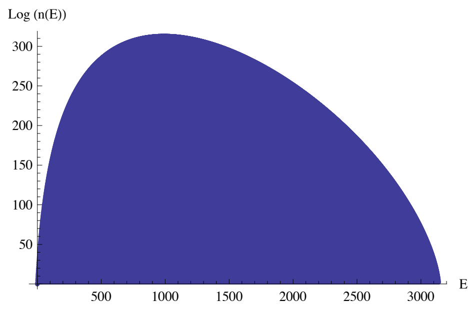
(a) Log-Density for a Clique problem *brock400_2.clq.cnf* from MaxSAT-2009.



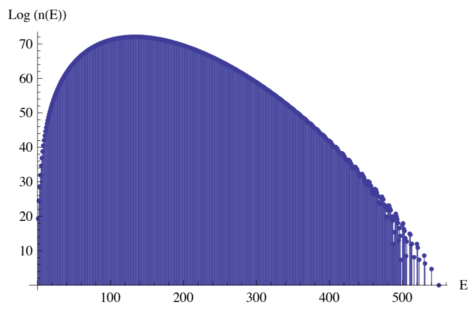
(b) Log-Density for a Spin Glass problem *spinglass5_10.pm3.cnf* from MaxSAT-2009. Notice there are no configurations with an even number of unsatisfied clauses.



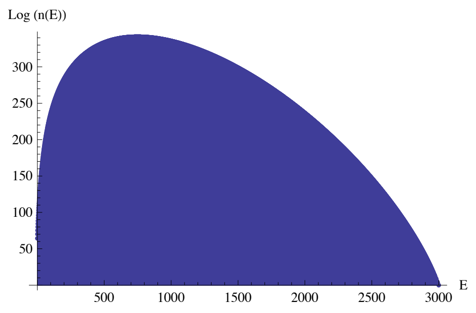
(c) Log-Density for a Clique problem *MANN_a27.clq.cnf* from MaxSAT-2009. No solver presented at MAXSAT09 could solve this instance (within 30 minutes).



(d) Log-Density for the Logistic problem *bw_large.a.cnf* from SATLib.



(e) Log-Density for the Pigeon Hole problem instance *hole10.cnf* from SATLib.



(f) Log-Density for the Morphed Graph Colouring problem *sw100-1.cnf* from SATLib. .

**Figure 5 Density of states (DOS) for several large formulas from MaxSAT-2009 and SATLib. E is the energy or number of unsatisfied clauses in the formula. N(E) is the number of models with E unsatisfied clauses.**

the subsets defined by the number of violated constraints, i.e., the number of truth assignments that violate exactly k constraints, for $0 \leq k \leq m$. Therefore, the problem of computing the density of states is a generalization of SAT, MAX-SAT and model counting

The additional information provided by the full density of states distribution is especially useful in the context of probabilistic models defined through combinatorial constraints such as Markov Logic Theories. In fact, the description of the state space can be used to efficiently compute not only the normalization constant of the underlying probabilistic model (also known as the partition function), but also its parameterized version. This level of abstraction is a fundamental advantage for learning methods because it allows us to reason about the system more abstractly. For example, in the case of a Markov Logic Theory, we can parameterize the partition function Z(w1, . . . , wK) according to the weights w1, . . . , wK of its K first order formulas that define the theory. Upon defining an appropriate energy function and obtaining the corresponding density of states, we can use the information about the partition function to directly compute the model parameters that best fit the training data.

To compute the density of states, we introduced MCMCFlatSat, a novel Markov Chain Monte Carlo (MCMC) sampling strategy, inspired by the Wang-Landau method ([6]), which is a flat histogram method from statistical physics. Given a combinatorial space and an energy function, a flat histogram method is a sampling strategy based on a Markov Chain that adaptively changes its transition probabilities until it converges to a steady state where it spends approximately the same amount of time in states with a low density of configurations (which are usually low energy states) as in states with a high density. This condition leads to a flat histogram of the energy levels visited that gives name to the method.

Technically, MCMCFlatSat is an Adaptive Markov Chain Monte Carlo method. In an Adaptive MCMC scheme, the transition probabilities are adjusted over time in order to achieve some optimality condition, learning the parameters while the chain runs. Even though it is usually harder to rigorously prove convergence properties, adaptive MCMC algorithms can significantly improve the performance over standard MCMC methods.

We conducted an extensive empirical analysis of MCMCFlatSat, demonstrating that our method converges quickly and accurately on a broad range of structured and synthetic instances. For instance, in the case of a logistic planning problem taken from SATLib, we are able to obtain this very fine grained information about a huge search space of $2^{459}$ assignments in a matter of minutes. Moreover, we showed that our method is remarkably precise, because it finds that there exists only one model (solution), but at the same time it is able to estimate the mode of the distribution, which is roughly $e^{300}$ times larger, thus counting both the needles and the haystack at the same time. See Figure 5.

Even if computing the entire density of states is a more general and more difficult problem than standard model counting, comparing MCMCFlatSat with model counters still provides some useful insights. In particular, we can show that when the number of constraints is not too big, that is, the overhead derived from computing the entire density of states is not overwhelming, MCMCFlatSat competes against state of the art model counters in terms of running times, and often provides more accurate estimates.

Because of the generality and the effectiveness of the flat histogram idea, we expect that this approach will find many other applications both in counting, probabilistic inference and learning problems.

### 3.2 Solution Clusters in Combinatorial Problems: Exact and Approximate Inference Methods

(See [14 and 3 ] for a detailed description of this work and background literature.)

Message passing algorithms, in particular Belief Propagation (BP), have been very successful in efficiently computing interesting properties of succinctly represented large spaces, such as joint probability distributions. Recently, these techniques have also been applied to compute properties of discrete spaces, in particular, properties of the space of solutions of combinatorial problems. For example, for propositional satsfiability (SAT) and graph coloring (COL) problems, marginal probability information about the uniform distribution over solutions (or similar combinatorial objects) has been the key ingredient in the success of BP-like algorithms. Most notably, the survey propagation (SP) algorithm utilizes this information to solve very large hard random instances of these problems.

Earlier work on random ensembles of Constraint Satisfaction Problems (CSPs) has shown that the computationally hardest instances occur near phase boundaries, where instances go from having many globally satisfying solutions to having no solution at all (a 'solution-focused' picture). In recent years, this picture has been redefined and it was found that a key factor in determining the hardness of instances in terms of search algorithm (or sampling algorithm) is the question: how are the solutions spatially distributed within the search space? This has made the structure of the solution space in terms of its clustering properties a key factor in determining the performance of combinatorial search methods (a 'cluster-focused' picture).

Can BP-like algorithms be used to provide such cluster-focused information? For example, how many clusters are there in a solution space? How big are the clusters? How are they organized? Answers to such questions will shed further light into our understanding of these hard combinatorial problems and lead to better algorithmic approaches for reasoning about them, be it for finding one solution or answering queries of probabilistic inference about the set of solutions. The study of the solution space geometry has indeed been the focus of a number of recent papers, especially by the statistical physics community, which has developed extensive theoretical tools to analyze such spaces under certain structural assumptions and large size limits.

We developed a purely combinatorial method for counting the number of clusters, which is applicable even to small size problems and can be approximated very well by message passing techniques (part of the work presented at the NIPS-08 conference; extended work in preparation for submission to PNAS). We proposed one of the first scalable methods for estimating the number of clusters of solutions of satisfiabilty (SAT) and graph coloring (COL) problems using a BP-like algorithm. While the naïve method, based on enumeration of solutions and pairwise distances, scales to graph coloring problems with 50 or so nodes and a recently proposed local search based method provides estimates up to a few hundred node graphs, our approach based on BP easily provided fast estimates for graphs with 100,000 nodes.

We validated the accuracy of the approach by also providing a fairly non-trivial exact counting method for clusters, utilizing advanced knowledge compilation techniques (BDDs and the DNNF representation). Our approach works with the factor graph representation of the underlying problem. We derived a 'partition function' style quantity, denoted Z(-1), to count the number of clusters; this quantity is formally proved to be exactly the number of clusters on 2-SAT and on 3-COL instances satisfying a certain simple graph property, and empirically found to be very close to exact on a number of structure and random instances. We then used the variational method to obtain BP equations for estimating Z(-1), the accuracy of which is validated independently. Overall, this approach provides a clear, principled method of reasoning about solution clusters of arbitrary discrete combinatorial problems. Unlike statistical physics based approaches such as survey propagation (SP), this approach starts with the first principles, precisely defining what 'clusters' are in the first place. The resulting BP(-1) equations for Z(-1) may be seen as an alternative and intuitive derivation of the much-studied SP algorithm for k-SAT, and for k-COL it provides a finer-grained methodology for reasoning about clusters than SP equations for that problem.

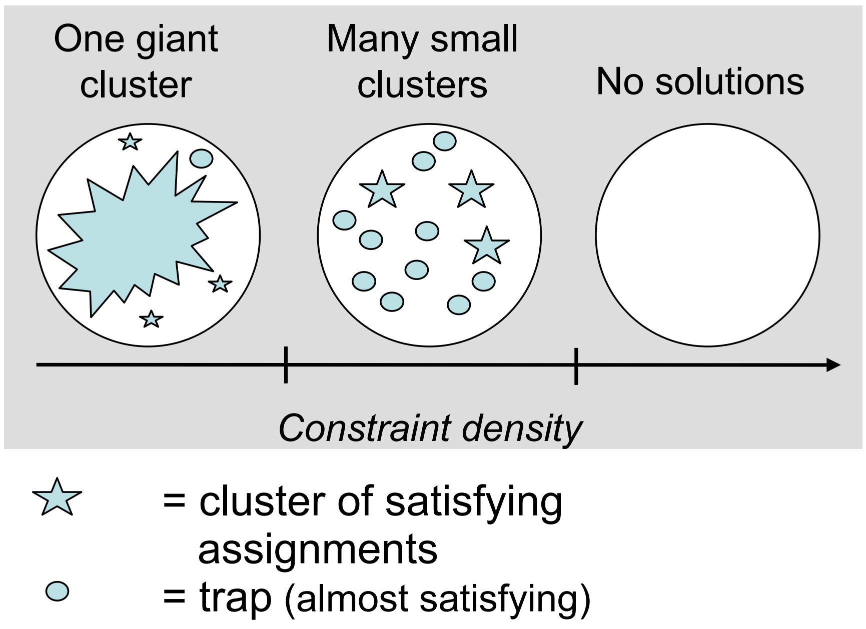


Figure 6 Pictorial representation of cluster regimes across different constrainedness regions

## 4. Uncovering hidden structure in combinatorial problems

[See [14 and 15 ] for a detailed description of this work and background literature.]

Capturing and exploiting problem structure is key to solving large real-world combinatorial problems. A very fruitful and prolific line of research that has been pursued in the study of combinatorial problems is the identification of various structural properties of instances that lead to efficient algorithms. Ideally, one prefers structural properties that are "easily" identifiable, such as topological properties of the underlying constraint graph. As an example, the degree of acyclicity of a constraint graph, measured using various graph width parameters, plays an important role with respect to the identification of tractable instances. Other useful structural properties consider the nature of the constraints, such as their so-called functionality, monotonicity, and row convexity.

Another approach for studying the nature of combinatorial problems of interest focuses on the role of *hidden* structure. One example of such hidden structure is a backdoor set—a set of variables B such that once they are instantiated, the remaining problem simplifies to a tractable class (but not necessarily syntactically defined). The notion of simplification or tractability in the definition of backdoor sets is captured by a polynomial time algorithm or sub-solver that, given a formula, either correctly decides its satisfiability or rejects it. This easily captures the behavior of the propagation procedures of the standard DPLL algorithm for backtrack search such as unit propagation and pure literal elimination. Note that the problem may become simple due to different reasons for different value assignments to the backdoor variables. Moreover, the actual semantics of the constraints may play a critical role in making the problem simple w.r.t. the sub-solver under consideration. These two aspects embedded in the notion of tractability through backdoors make this kind of structure "hidden", in contrast to other structural notions such as bounded tree-width of the underlying constraint graph.

The understanding of backdoor sets has important practical implications. A combinatorial problem with n variables and a backdoor set B can be solved by considering only $2^{|B|}$ variable assignments instead of all $2^n$ variable assignments (in the worst case), thereby yielding considerable computational savings when the backdoor set is a small subset of the set of all n variables. Therefore, the notion of a small backdoor set succinctly capturing the combinatorics of a problem provides a tool for analyzing and understanding the efficiency and performance of state-of-the-art solution techniques for large-scale real-world combinatorial problems. In addition, the demonstration of the existence of very small backdoor sets in real-world combinatorial problems has contributed to the design of novel search techniques by motivating the use of randomization, restarts, and algorithm portfolios in existing solution approaches.

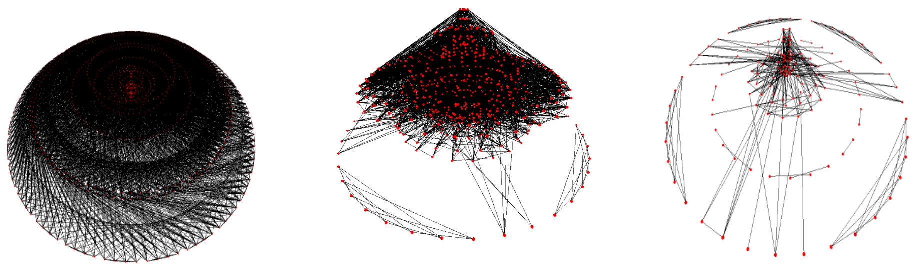


**Figure 7 Constraint graph of a real-world instance from the logistics planning domain. The instance in the plot has 843 vars and 7,301 clauses. One backdoor set for this instance w.r.t. unit propagation has size 16 (not necessarily the minimum backdoor set). Left: Constraint graph of the original instance. Center: Constraint graph after setting 5 variables and performing unit propagation. Right: Constraint graph after setting 14 variables and performing unit propagation.**

From a practical point of view, the usefulness of backdoor sets depends on two factors: 1) whether problems of interest indeed have small backdoor sets, and 2) whether such small backdoor sets can be identified efficiently. In our research we compared different backdoor variants highlighting an important tradeoff between the size of the smallest backdoor and the computational complexity of deciding the existence of a backdoor set of a given size. We provided both theoretical and empirical characterizations of such tradeoffs.

We examined the notion of backdoor sets in the context of Boolean satisfiability (SAT). Most complete search procedures for SAT are based on the Davis–Putnam–Logemann–Loveland (DPLL) algorithm, a backtrack search procedure where one systematically chooses the next variable to assign and then applies polynomial-time propagation procedures (or sub-solvers) such as unit propagation and pure literal elimination to infer as many additional assignments as possible. The original work on backdoors for SAT was done with these kinds of polynomial-time algorithms in mind. Since then, follow-up work has focused on backdoor sets for which the resulting simplified sub-problems belong to a well-understood syntactically defined tractable class of conjunctive normal form (CNF) formulas, such as Horn, 2CNF, or renamable Horn (RHorn). Nishimura, Ragde, and Szeider introduced the notion of so-called "deletion" backdoors with respect to the syntactic tractable classes Horn and 2CNF, where deleting all occurrences of the backdoor variables from the formula results in a subformula that is 2CNF or Horn, respectively. This differs from the original notion of a strong backdoor set, where one needs to consider all possible value assignments to the backdoor variables. Nishimura, Ragde, and Szeider showed that deletion backdoor sets w.r.t. Horn and 2CNF can be found efficiently and that deletion backdoors and strong backdoors w.r.t. Horn and 2CNF are in fact equivalent. These positive formal results have motivated work on backdoors w.r.t. the tractable class of RHorn formulas, which is a strict superset of the class of all Horn formulas. First, several heuristic approaches for finding small deletion RHorn-backdoors were proposed. Later, Razgon and O'Sullivan showed formally that the existence of a deletion RHorn-backdoor of a fixed size can be decided in polynomial time.

Theoretically, we showed that the usefulness of deletion RHorn-backdoors is limited—they can be exponentially larger than the smallest strong RHorn-backdoors. Although backdoors w.r.t. the syntactic tractable classes 2CNF, Horn, and RHorn have been the subject of numerous theoretical papers showing some positive complexity results, empirical evidence that real-world problems in fact have small backdoors w.r.t. these classes is lacking. We provided integer programming encodings for finding the smallest deletion Horn- and RHorn-backdoors and empirically evaluate the size of the small- est deletion backdoors for these classes. Our results on a set of benchmarks show that the smallest deletion backdoors with respect to these well-understood tractable classes are consistently considerably larger than strong backdoors with respect to DPLL sub- solvers such as unit propagation and "probing" (also known as the failed-literal rule). For example, on a set of graph-coloring instances, probing results in backdoors of size less than 0.33% of the total number of variables, while the smallest deletion Horn- backdoors contain 66.67% of the variables. Our formal and empirical findings highlight the tradeoff between the favorable complexity of finding deletion 2CNF-, Horn-, and RHorn-backdoors and the large size of the smallest such backdoors in practice.

One key property of polynomial-time algorithmic sub-solvers employed by state-of- the-art SAT solvers is the detection of trivially inconsistent formulas, that is, formulas that contain an empty clause. This property is not considered for tractable classes such as 2CNF, Horn and RHorn. To address this issue, we defined the larger tractable class 2CNF{} as the class of formulas that includes all 2CNF formulas as well as all formulas that contain an empty clause, and we defined the tractable classes Horn{} and RHorn{} similarly. Accounting for the presence of an empty clause may seem like an inconsequential feature for a tractable class or a polynomial-time sub-solver underlying a backdoor set. However, we showed that including empty-clause detection can dramatically reduce the size of the resulting backdoor sets, albeit at the cost of increasing the worst-case complexity of backdoor detection beyond the "within NP" results known for the pure classes 2CNF, Horn, and RHorn. More precisely, we proved that deciding whether a given formula has a strong 2CNF{}-, Horn{}-, or RHorn{}-backdoor of fixed size k is both NP- and coNP-hard, and therefore strictly harder than NP, assuming NP $\neq$ coNP. However in terms of backdoor size, we showed that there exist families of formulas for which considering the tractable classes 2CNF{}, Horn{}, and RHorn{} leads to arbitrarily smaller backdoors than backdoors w.r.t. the pure 2CNF, Horn, and RHorn classes, respectively. In addition, empirically we found that in certain graph-coloring instances with planted cliques of size 4, while the smallest strong Horn-backdoor sets involve two-thirds of the variables, the fraction of variables in the smallest strong back- doors with respect to mere empty-clause detection converges to 0 as the size of the graph grows. These results again highlight the tradeoff, as a function of the underlying tractable class, between the size of the smallest backdoor set and the computational complexity of deciding the existence of a backdoor set of a given size. Our work characterizing the different variants of backdoor sets, both in size computational complexity, provides interesting insights into the development of new solution methods, which exploit structure in real-world instances.

The original definition of a strong backdoor set B captures the fact that a systematic tree search procedure (such as the DPLL procedure for SAT) restricted to branching only on variables in B will successfully solve the problem. Furthermore, the tree-search procedure restricted to branching on the variables in B will succeed independently of the order in which it explores various parts of the search tree. However, most of the state-of-the-art DPLL-based SAT solvers, in addition to using sophisticated branching heuristics and data structures, rely heavily on clause learning, that is, adding new constraints or "nogoods" every time a conflict is encountered during the tree search. Clause learning is extremely useful in practice in addition to enabling provably exponentially shorter proofs of unsatisfiability.

Adding new information as the search progresses has, however, not been considered in the traditional concept of backdoors. To address this limitation, we formally extended the concept of backdoors to the context of learning, where the branching order over the backdoor variables is taken into account and information learned from previous search branches is used by the sub-solver underlying the backdoor. The extended notion often leads to much smaller backdoors than the "traditional" ones. In particular, we proved that the smallest backdoors for SAT that take clause learning into account can be exponentially smaller than traditional backdoors that are oblivious to this solver feature. We presented empirical results showing that the added power of "learning-sensitive backdoors" is observable in practice by comparing backdoor sizes with and without clause learning for a set of real-world problems.

Historically, there have been many similarities between research on combinatorial decision problems—in particular, Boolean satisfiability (SAT) – and research on combinatorial optimization problems – in particular, mixed-integer linear programming (MILP). These similarities suggest that concepts that have been used successfully in one realm can perhaps be extended to the other realm and lead to new insights. We investigated this from the angle of applying ideas from SAT to MILP. We extended the concept of backdoor sets to optimization problems, which raises interesting new issues not addressed by earlier work on backdoor sets for satisfiability. We introduced "weak optimality backdoors" for finding optimal solutions and "optimality-proof backdoors" for proving optimality. Similarly to clause learning in satisfiability search methods, effective optimization algorithms often involve adding new information such as cuts and tightened bounds as the search progresses. Therefore, we also introduced "learning-sensitive" backdoors for optimization.

We provided the first experimental results showing that small backdoor sets exist for benchmark instances of mixed-integer linear programming optimization problems, and found that such instances often have backdoors involving fewer than 5% of the discrete variables. In addition, we demonstrated that studying backdoor distributions – capturing the probability that a random subset of the set of all the variables is a backdoor set as a function of the size of the subset – gives insight into search behavior. One prefers a backdoor distribution where subsets of small size have high probability of serving as backdoor sets. We provided empirical evidence that, for a given problem, the quality of the distribution of weak optimality backdoors relative to that of optimality-proof backdoors aligns roughly with the quality of the runtime distribution when finding an optimal solution, relative to the quality of the runtime distribution when proving optimality. Finally, we also designed a simple heuristic for selecting backdoor variables based on information provided by linear programming relaxations and showed that it can be used effectively when searching for small backdoors.

## 5. Combinatorial Optimization for Over-Constrained Problems

(See [12 and 13] for a detailed description of this work and background literature.)

We have proposed a new hybrid strategy for optimizing over-constrained discrete combinatorial problems, where simultaneously satisfying all constraints is impossible and the goal is to find a value assignment to variables that satisfies as many constraints as possible. In the context of SAT, this is referred to as the Maximum Satisfiability or the MaxSAT problem. The proposed method combines the complementary strength of local search and systematic search, bringing the best of both worlds in a way that is ideal for multi-core architectures.

Combining local and systematic search methods in a fruitful way has been a challenge, with limited success so far. The main bottleneck has been the nature and cost of information exchange between the solvers. Key design decisions include: what kind of information to communicate (lightweight or heavyweight), how to communicate it (message passing and synchronization or shared memory), how to use the communicated information (strict guidance or soft guidance), etc. Hybrid solvers are often designed so that one solver waits for the other to finish during various stages of the search, thereby reducing the overall efficiency.

Our proposed technique, based on shared memory architecture, enables continuous information exchange between two constraint solvers without slowing down either of the two. The main search effort here is driven by a local search algorithm, which is loosely coupled with and guided by a systematic search algorithm. Such a hybrid search strategy is surprisingly effective, leading to substantially better quality solutions to many challenging MaxSAT instances than what the current best exact or heuristic methods yield, and it often achieves this within seconds. This hybrid approach is naturally best suited to MaxSAT instances for which proving unsatisfiability is already hard; otherwise the systematic solver finishes a little too early and the method falls back to pure local search. Experiments on a large suite of hard, infeasible, industrial 'real-world' instances from the SAT Race 2008 competition have revealed a unique search behavior of the hybrid approach, and surprisingly good results by the solver, called MiniWalk, on nearly all of the instances considered. Unlike usual local search methods, which slowly but often uniformly move closer towards a solution with some noise, the hybrid method appears to stay fairly far from solutions most of the time during the search but every once in a while makes very steep descents towards a solution, presumably guided by the new search space areas that the coupled systematic search has moved to. Such a search behavior has not been observed before for local search methods.

We have also explored a complementary direction (presented at the SAT-09 conference), where the main search effort for an optimal solution is guided by a 'relaxed' systematic solution finder, and the final candidate solution is further improved in quality using local search. Systematic search solvers typically work using a branch-and-backtrack scheme – fixing values of variables one at a time, testing whether a contradiction is detected by constraint propagation, and if so, backtracking to flip the value of the nearest conflicting variable.

Constraint propagation and conflict analysis schemes, such as unit propagation and first unique implication point, are key to the success and scalability of Boolean satisfiability solvers. However, these techniques are not logically sound for MaxSAT style optimization problems. The team has proposed a relaxation of the systematic search paradigm as a heuristic method to find very good quality (though not necessarily optimal) solutions, which are then improved further using local search. This relaxation brings the power of constraint propagation and conflict analysis to MaxSAT solvers, as a heuristic strategy, while extending the solver to tolerate a small pre-specified number of conflicts. The resulting solver, called RelaxedMinisat, is the only (MaxSAT) solver capable of identifying a single bottleneck constraint in all but one instance in a test suite consisting of all unsatisfiable SAT Race 2008 industrial instances.

## 6. Multiagent gaussian inference for dynamic processes

(See [2 and 4] for a detailed description of this work and background literature.)

Distributed inference tasks are becoming more and more important as myriads of tiny inexpensive sensing devices are being deployed, such as in phones and building materials. Problems of this type occur in a variety of different applications, ranging from multi-robot systems to wireless sensor networks, and include tracking, environmental and habitat monitoring, smart buildings control and surveillance activities. Despite the application specific differences, many of these inference problems can be modeled as a network of sensing devices that can perform local computations and communicate with other nodes, collaborating to produce global information from individual local data.

In many of the settings mentioned, a centralized solution, in which a single computational node receives and processes all the information available, is either not feasible due to communication and energy restrictions or not desirable because it introduces a single point of failure and additional delays. Therefore there is a need for distributed solutions where inference is performed locally at each node on the basis of information that is retrieved both locally and by communication with neighboring nodes.

The focus of our work concerning this topic has been on the general problem of Bayesian estimation, where a probability model is assumed to be known and one is interested in computing the posterior distribution of a collection of hidden variables ("the state"), given the evidence collected by a network of sensing devices. In particular, we focused on dynamic scenarios where the state of the world is changing over time.

An example of such a problem is tracking, where the position (or its probability distribution) of an object moving in a sensor field needs to be estimated on the basis of the noisy measurements collected by a network of sensing devices.

The main contribution of our work is a new framework to study distributed estimation problems in dynamic settings based on graphical models. Our approach generalizes the derivation of the Kalman filter in terms of Belief Propagation to a distributed setting. Using our framework, we obtained novel distributed estimation algorithms based on message passing techniques where each node is able to locally elaborate and fuse the information it receives before transmitting it again, thus distributing the computational burden and also reducing the use of communication resources.

We evaluated our solution on a tracking application where the goal is to estimate the position of a moving target. We showed in simulation that our method outperforms the other state of the art techniques in terms of estimation error on a general class of problems, even in presence of data loss. Moreover, we showed that our solution is close to the theoretical optimum that is achievable in the presence of communication latencies.

## 7. Summary

This research effort focused on three core research challenges: (1) How to explain the gap between formal analysis and practical performance for combinatorial search; (2) How to characterize and capture hidden tractable structure in real-world problems; and, (3) How to further boost combinatorial search methods for real-world problems. Predicting the runtime of combinatorial search methods is a notoriously hard problem due to tremendous variations in runtime observed when solving practical problem instances. We developed a series of advanced formal models for predicting the runtime of combinatorial search methods. We introduced different models of runtime distributions of search methods capturing exponential and power law (heavy-tailed) regimes for both complete and incomplete randomized search methods, together with a generative model that generates search trees with any pre-defined degree of heavy-tailedness. In order to better understand and model solution spaces of combinatorial problems, we also developed new methods for the efficient computation of the number solution clusters and their marginal distributions. These methods can effectively handle practical problem instances with tens of thousands of variables, containing solution clusters with sizes ranging over many orders of magnitude. Reasoning based on such clusters has been the key component of highly successful combinatorial search methods proposed recently. We extended the notion of “backdoor sets,” --- a measure that characterizes hidden problem structure --- to encompass combinatorial optimization problems as well as learning during search, thereby providing novel insights into the connection between hidden structure of optimization problems and the surprising efficiency of today’s optimization engines. We also analyzed probabilistic reasoning techniques based on message passing, namely belief propagation and survey propagation, in the context of combinatorial problems in the Boolean satisfiability domain, resulting in the first detailed study of the evolution of these search methods over time as well as the utilization of these techniques to provide statistical estimates on key properties of the solution space. In addition we studied the problem of computing the density of states of a Boolean formula, which is a generalization of Satisfiability Testing, MAX-SAT, and model counting, and developed a novel Markov Chain Monte Carlo sampling strategy, inspired by a flat histogram method from statistical physics. The new sampling method provides novel insights into combinatorial search spaces that lie far beyond the reach of previous techniques. We also formulated multi-agent inference problems in dynamic environments into the framework of message passing algorithms and graphical models, generalizing to the distributed case of the Kalman filter. We proposed a new hybrid strategy for optimizing the MaxSAT problem, combining the complementary strength of local search and systematic search, bringing the best of both worlds in a way that is ideal for current multi-core architectures.

**References**

We list here archival publications that describe the research supported by this grant. Note that refereed conference proceedings are the preferred academic outlet in computer science.

1. Stefano Ermon, Carla Gomes, and Bart Selman. A Flat Histogram Method for Computing the Density of States of Combinatorial Problems. *Proc. 22nd International Joint Conference on Artificial Intelligence (IJCAI*), July 2011.

2. Stefano Ermon, Carla Gomes, and Bart Selman. A Message Passing Approach to Multiagent Gaussian Inference for Dynamic Processes. *Proc. 10th International Conference on Autonomous Agents and Multiagent Systems* (AAMAS), May 2011.

3. Ermon, Stefano; Gomes, Carla; and Selman, Bart. Computing the Density of States of Boolean Formulas. *Proc. of the 16th International Conference on Principles and Practice of Constraint Programming* (CP-10), 2010. (Best student paper award.)

4. Stefano Ermon, Carla Gomes, and Bart Selman. Collaborative Multiagent Gaussian Inference in a Dynamic Environment Using Belief Propagation. *Proc. 9th International Conference on Autonomous Agents and Multiagent Systems* (AAMAS), 2010.

5. Lukas Kroc, Ashish Sabharwal, Bart Selman. An Empirical Study of Optimal Noise and Runtime Distributions in Local Search. *Proc. of the 13th International Conference on Theory and Applications of Satisfiability Testing*, (SAT-2010) LNCS volume 6175, pp 346-351, Edinburgh, UK, July 2010.

6. Ahmadizadeh, Kiyan; Dilkina, Bistra; Gomes, Carla; and Sabharwal, Ashish. An empirical study of optimization for maximizing diffusion in networks. *Proc. of the 16th International Conference on Principles and Practice of Constraint Programming* (CP-10), 2010.

7. Sheldon, Daniel; Dilkina, Bistra; Elmachtoub, Adam; Finseth, Ryan; Sabharwal, Ashish; Conrad, Jon; Gomes, Carla; Shmoys, David; Allen, Will; Amundsen, Ole; and Vaughan, William. Maximizing Spread of Cascades Using Network Design. *Proc. of the 26th Conference on Uncertainty in Artificial Intelligence (UAI-10), 2010.*

8. Ansotegui, Carlos; Bejar, Ramon; Fernandez, Cesar; Gomes, Carla; and Mateu, Carles. Generating Highly Balanced Sudoku Problems as Hard Problems. *Journal of Heuristics,* Volume 16, 2010.

9. Carvalho, Alda; Crato, Nuno; and Gomes, Carla. A generative power-law search tree model. *Computer and Operations Research*, Volume 36, 2376–2386, 2009.

10. Guo, Yunsong and Gomes, Carla. Learning Optimal Subsets with Implicit User Preferences. *Proc. of the 21st International Joint Conference on Artificial Intelligence* (IJCAI-09), 2009.

11. Guo, Yunsong and Gomes, Carla. Ranking structured documents: a large margin based approach for patent prior art search. *Proc.of the 21st International Joint Conference on Artificial Intelligence* (IJCAI-09), 2009.

12. Kroc, Lucas; Sabharwal, Ashish; Selman, Bart; and Gomes, Carla. Integrating Systematic and Local Search Paradigms. *Proc. of the 21st International Joint Conference on Artificial Intelligence* (IJCAI-09), 2009.

13. Lukas Kroc, Ashish Sabharwal, Bart Selman. SAT-09. Relaxed DPLL Search for MaxSAT. *Proc. 12th International Conference on Theory and Applications of Satisfiability Testing*, LNCS volume 5584, pp 447-452, Swansea, Wales, U.K., June 2009.

14. Dilkina, Bistra; Gomes, Carla; and Sabharwal, Ashish. Backdoors in the Context of Learning. *Proc. of the 12th International Conference on Theory and Applications of Satisfiability Testing* (SAT-09), 2009.

15. Dilkina, Bistra; Gomes, Carla; Malitski, Yuri; Sabharwal, Ashish; and Sellmann, Meinolf. Backdoors to Combinatorial Optimization: Feasibility and Optimality. *Proc. of the 6th International Conference on Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems* (CPAIOR-09), 2009.

16. Lukas Kroc, Ashish Sabharwal, Bart Selman. Counting Solution Clusters in Graph Coloring Problems Using Belief Propagation. *Proc. of the 22nd Annual Conference on Neural Information Processing Systems* (NIPS-08), pp 873-880, Vancouver, BC, Canada, Dec 2008.