Reliable Cost Predictions for Finding Optimal Solutions to LABS Problem: Evolutionary and Alternative Algorithms

Franc Brglez[1], Xiao Yu Li[1], Matthias F. Stallmann[1] and Burkhard Militzer[2]

[1]Computer Science Department, NC State University, Raleigh, NC 27695, USA

[2]Lawrence Livermore National Laboratory, Livermore, CA 94550, USA

ABSTRACT

The *low-autocorrelation binary sequence* (LABS) problem represents a major challenge to all search algorithms, with the evolutionary algorithms claiming the best results so far. However, the termination criteria for these types of stochastic algorithms are not well-defined and no claims have been made about optimality. Our approach to find the optima of the LABS problem is based on (1) experiments with problem sizes for which optimal solutions are known, (2) an asymptotic analysis of statistics generated by such experiments, (3) reliable predictions of the cost required to find optimal solutions for larger problem sizes. The proposed methodology provides a well-defined termination criterion for evolutionary and alternative search algorithms alike.

Keywords: fundamental combinatorial algorithms, performance predictions and evaluations, low-autocorrelation binary sequence, evolutionary algorithms

1. INTRODUCTION

The *low-autocorrelation binary sequence* (*LABS*) problem has a simple formulation. Consider a binary sequence of length L, $S = s_1 s_2 \ldots s_L$ and their autocorrelations $C_k(S) = \sum_{i=1}^{L-k} s_i s_{i+k}, s_i = \{+1, -1\}$, for $k = 1, \ldots, L-1$. The energy function of interest is $E(S) = \sum_{k=1}^{L-1} C_k^2(S)$ which defines the *merit factor F* of the sequence [1]: $F = L^2/(2E)$.

The objective of optimization is to find a sequence that maximizes F, or equivalently, minimizes E. Finding an optimum sequence has important applications in communication engineering and is also of interest to physicists since the sequence models one-dimensional systems of Ising-spins [2].

The asymptotic value for the maximum merit factor F is known [1] and has also been re-derived using arguments from statistical mechanics [2]: as $L \to \infty$, $F \to 12.3248$. In Figure 1 we superimpose the asymptotic limit, the known optimal merit factors for $L \leq 60$ from exact search algorithms [3, 4], the best known merit factors for larger L from stochastic algorithms [5, 6, 7, 8, 9], and also from this work. We observe that (1) the exact factors, limited to values of $L \leq 60$, do tend to follow the projected trend, (2) the factors reported by current heuristics are clearly diverging from the asymptotic value as $L > 100$, and (3) the factors reported by the heuristic in this paper do support the trend established by the exact method. While on the scale shown this may not seem much, the minimum energy levels reported in Figure 3(b) point to merit factors that are on par with the ones reported only recently [9]. For example, given $E_{min} = 208$ for $L = 64$ in Figure 3(b), we find $F = 9.84615$. Note also
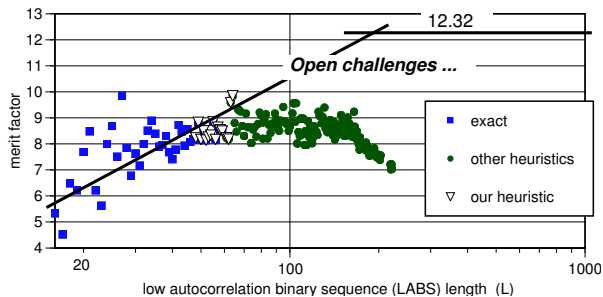

Figure 1: The divergence of earlier LABS heuristics.

that merit factors reported in [5, 6, 7, 8], are significantly better than the solutions based on simulated annealing in [2].

The challenge of finding *exact minima* to the LABS problem can be best appreciated in terms of the runtime complexity needed to solve the problem by the current generation of algorithms. The branch-and-bound algorithm devised and implemented by Mertens [3] reports runtime complexity of $O(1.85^L)$ and a total runtime of '313 hours of CPU time on a four-processor Sun SPARCstation 20' to find optimal merit factors for $L \leq 48$. During 2003, additional optimal merit factors were posted for $L \leq 60$ [4]. An alternative algorithm, using a stochastic approach, reports a runtime complexity of $O(1.68^L)$ [8]. However, while being able to replicate the known minima for $L \leq 48$, the best value reported for $L = 61$ is $E_{min} = 246$, relatively far from 226 shown in Figure 3(b).

Our approach to finding minima of the LABS problem relies on two generic concepts [10]:

- Termination criterion 'A'. We use the known LABS minimum value for a given L to terminate a local search algorithm when the optimal sequence is found for the first time. Variables recorded during this search include, at the minimum, *runtime* and *samples* (the number of cost function evaluations) that are required to find the optimum. Both are random variables for which we find statistics so we can deduce the *probability of success*, P_{succ}, of finding the LABS minimum value, given the *runtime* or *samples* constraints.

- Termination criterion 'B'. Results of the preceding stage allow us to *predict*, for a given value of L (for which the minimum is unknown), the *required runtime* and *required samples* so that we may find the minimum with the probability of success P_{succ}. The *required samples* must be used as the *termination criterion 'B'* if experiments are executed on a platform different from the one used under the *termination criterion 'A'*.

The paper is organized as follows. Section 2 defines the experimental setup and reports on LABS problem experiments under the termination criterion 'A', using the Kernighan-Lin solver (KL) [11, 12, 13]. Section 3 relies on data from preceding section to report experimental results under the termination criterion 'B', not only replicating the minima from [3, 4] at a significantly lower computational cost but also introducing new minima for $L = 61, 62, 63, 64$. Section 4 evaluates the asymptotic performances of the KL solver and an ES solver which is based on the evolutionary strategies algorithm [7], leading to new insights and conclusions.

2. TERMINATION CRITERION 'A'

The experimental testbed we use is outlined in Figure 2. Each experiment is initialized with a unique random seed from the file *randomTriplets* that contains three random integers on each line, making the testbed environment cross-platform consistent for any LABS solver that uses the standard IEEE random generator. Termination of each execution is controlled by the known optimum value stored in the file *knownOptima*, available at [4].

Data of most interest generated by these experiments are *runtime* (seconds, platform-specific) and *samples* (total number of samples to evaluate the cost function, platform-independent). These two random variables are closely correlated and, not surprisingly, both will have *exponential distribution* for stochastic solvers such as KL or EA described later in this paper.[1] While the exponential distribution of *runtime* or *samples* implies significant variability as illustrated in Figure 2, the variability is also *predictable* – shown by the close fit of observed and theoretical distributions. For example, if we allow the KL solver to search for the optimum sequence of length $L = 34$ for 3,088,247 samples (from *any randomly choosen initial sequence*), the probability of finding the optimum is 0.632. However, if we increase the search limit to 4*3,088,247 samples or 8*3,088,247 samples, the probability increases to 0.981 or 0.999 respectively.

Our implementation of KL solver is based on the Kernighan-Lin algorithm [11, 12, 13] and it samples the LABS cost function in a sequence of well-defined *moves*, starting from a randomly chosen initial configuration. If no further improvement can be gained and the known optimum is not reached, the solver *restarts* from another randomly chosen configuration. Just like *runtime* and *samples*, *moves* and *restarts* both have exponential distribution. To capture the asymptotic behavior of the KL solver, we performed 128 experiments for each value of L ranging from 10 to 47. The asymptotic behavior of average *restarts* is shown in Figure 3; the average *runtime* and *samples* are shown in Figure 4. We analyze average *restarts* in the next section.

3. TERMINATION CRITERION 'B'

We ask the question: 'how long should the KL solver run in order to find the optimum for $L > 47$ with a given probability of success P_{succ}?' We find the answer by analyzing the asymptotic behavior of either *runtime*, *samples* or *restarts* of the KL solver under termination criterion 'A'. An example of data is shown in the top part of Figure 3. The average

---

[1]Exponential distribution of runtime has been observed not only for stochastic solvers in different contexts [14], but also within a unified framework for stochastic *and* branch-and-bound solvers applied to SAT problem instances [15]. Distributions other than exponential have also been observed in a number of related experiments with several solvers [15].

Experiments under termination 'A'

▷ Create *LABSbed* as a LABS-problem specific testbed environment, simpler than but similar to *SATbed* [16]. We consider (1) primary input files *randomTriplets* and *knownOptima*, (2) user-specified parameters L, *solverList*, *nExperiments*, (3) a solver encapsulator SOLVERENCAP_A(L, *randomTriplets*, *knownOptima*, *solverList*, *nExperiments*) that returns *rawData*(*solver*, L), and (4) GETSTATS(*rawData*(*solver*, L)) that returns *statsData*(*solver*, L).

▷ Invoke SOLVERENCAP_A to return a tabular report *rawData*(*solver*, L) for each solver and each L, with data columns under labels such as instanceNum, runtime, samples, solution. The number of rows is determined by *nExperiments*. For the same value of L, each experiment represents an equivalent problem instance, initialized by the LABS solver with a different random seed triple from the file *randomTriplets*.

▷ Invoke GETSTATS to return various statistics for random variables in each data column in *rawData*(*solver*, L), including characterization of underlying distributions and the respective solvability functions [16]. As shown below, the number of samples (or function evaluations) required by the KL solver to find the optimum solution for $L = 34$ for *each* of the 128 experiments has exponential distribution and can vary dramatically. At best, an optimum solution is found with 43,354 samples; at worst with 13,322,800 samples; with a median, mean and standard deviation of 2,115,035, 3,088,247, and 3,006,546 samples, respectively. The runtime mean value, specific to our platform (a Linux PC with 266 MHz processor), is 6.60 seconds. Hyperlinks to complete raw and statistical results of experiments such as illustrated here, covering several solvers, and values of L up to 64 can be found on the newly created *LABSbed* home page [17].
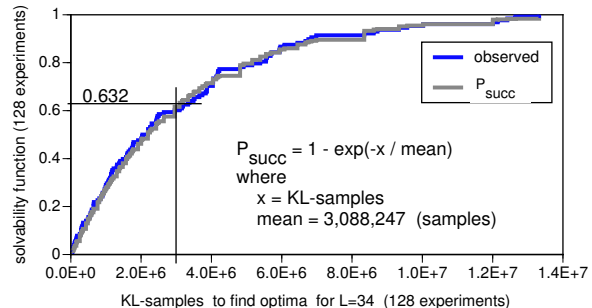

Figure 2: Experimental flow under termination 'A'.

values of restarts are based on 128 experiments for consecutive values of $L = 20$ to $L = 47$. Also shown are three regression lines derived from the observed data for $L \leq 47$ and data points as predicted by the respective regression lines for values of $L > 47$:

$$\text{nominal restarts predictor} = 0.1351 \times 1.331^L \quad (1)$$
$$\text{upper bound restarts predictor} = 0.6754 \times 1.331^L \quad (2)$$
$$\text{lower bound restarts predictor} = 0.0270 \times 1.331^L \quad (3)$$

According to the *nominal restarts predictor* in (1), the number of *required restarts* to find the optimum with $P_{succ} = 0.632$ for $L = 48, 60$, and 64, is 123,851 or 3,832,423 or 12,031,820, respectively. However, not all average restarts

reported by the KL-solver are 'on the line' that corresponds to the *nominal restarts predictor*. In fact, the *maximum deviations* from the nominal predictions for *required restarts* for the range of $20 \leq L \leq 47$ are very significant: for $L = 26$ the prediction is far too conservative (229 restarts predicted, 64 observed, a ratio of 3.59), for $L = 41$ the prediction is far too optimistic (67,375 restarts observed, 16,726 predicted, a ratio of 4.03). Such variations by far exceed the expected statistical variability of the observed mean values of restarts: the 95% confidence intervals for the observed mean values are [52 ... 77] for $L = 26$ and [55,527 ... 79,222] for $L = 41$, respectively. The empirical *upper bound restarts predictor* in equation (2) simply multiplies the *nominal restarts predictor* by a factor of 5 which is based both on the worst-case ratio of 4.03 and the observed confidence interval for mean restarts at $L = 41$. Using the *upper bound restarts predictor*, the number of *required restarts* for $L = 41$ has the value of 81,380. Similarly, the empirical *lower bound restarts predictor* divides the *nominal restarts predictor* by a factor of 5. Clearly, such bounds may need to be adjusted for larger values of L when more experimental data becomes available.

The effectiveness of these predictors as the termination criterion 'B' is demonstrated by a series of experiments with $48 \leq L \leq 64$ on a cluster of 4 workstations, summarized in the table in Figure 3. Using the *nominal restarts predictor* in equation (1), we performed 16 experiments for each $48 \leq L \leq 62$, 8 experiments for $L = 63$, and 4 experiments for $L = 64$. The columns reported in the table refer to sequence length (L), best known minimum (E_{best}), best minimum reported by KL solver (E_{min}), number of experiments ($nExpr$), number of times E_{min} is found ($nHits$), number of unique solutions reporting E_{min} ($uSols$), hit ratio of $nHits$ versus $nExpr$ ($hitR$), runtime required to execute each experiment ($hours$ on a 266 MHz workstation under Linux).

Assuming that the *nominal restarts predictor* values that terminate each run are 'correct', we expect each experiment to find the optimum with $P_{succ} = 0.632$, i.e. we expect all E_{min} to match or improve on E_{best} and the hit ratio to be 0.632. For 17 values of L in this exploratory study, we observe a hit ratio mean of 0.768 with a standard deviation of 0.191 – somewhat better than expected. The most likely reason is that for most values of L in this range, the true (and unknown) mean restart values are below the regression line. For example, finding 16 hits out of 16 experiments for $L = 60$ implies that the predicted mean value of 3,832,423 restarts should really be *reduced* by a factor of 3, 4, or 5.

Now, we cannot say that the values of E_{min} we report for for $L > 60$ are exact optima. However, given the experimental results at hand, the minima reported for 16 experiments with a hit ratio greater than or equal to 0.625 are the likely optima. If this threshold is not maintained, the number of restarts must be increased until we reach the threshold. To increase the overall reliability of the predictions, we can also increase the number of experiments while maintaining the hit ratio close to 0.632. Increasing the number of experiments for $L = 63, 64$ is in progress, with experiments for $L > 64$ to follow.

Compared to performance of earlier algorithms, the performance of the KL solver under the termination criterion B as shown in the graph and the last column of the table in Figure 3 is very effective: for $L = 60$ it took $16 \times 20 = 320$ hours to find not only one optimum solution but 16 solutions from which 10 optima were found to be unique. As shown in the graph, the *runtime* complexity of this solver is $O(1.42^L)$. The question arises, can we do better with an alternative solver. We return to the methodology introduced in Figure 2 to address this question in the next section.

(a) restarts predictions for the termination criterion 'B'
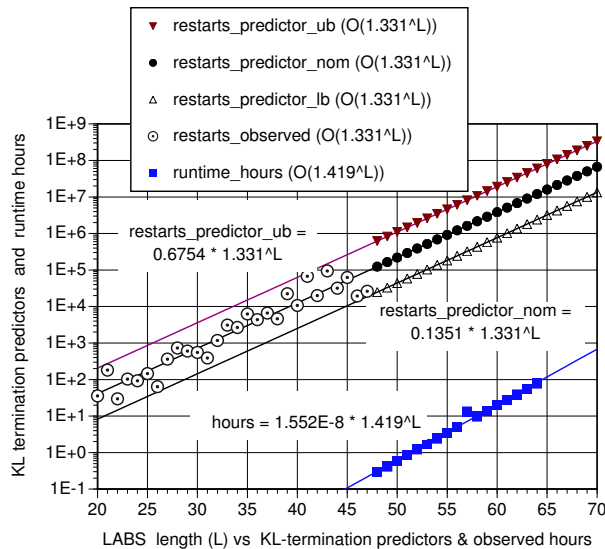
(b) summary of experiments based on predictions above

L	E_{best}^1	E_{min}	nExpr	nHits	uSols	hitR	hours[2]
48	140[4]	140	16	11	7	0.6875	0.3
49	136[4]	136	16	12	9	0.75	0.4
50	153[4]	153	16	15	11	0.9375	0.6
51	153[4]	153	16	5	3	0.3125	0.8
52	166[4]	166	16	12	7	0.75	1.2
53	170[4]	170	16	12	8	0.75	1.7
54	183[4]	183	16	10	6	0.625	2.4
55	171[4]	171	16	14	7	0.875	3.5
56	192[4]	192	16	16	7	1	5.0
57	188[4]	188	16	11	3	0.6875	13.2
58	197[4]	197	16	13	7	0.8125	9.8
59	205[4]	205	16	16	9	1	13.9
60	218[4]	218	16	16	10	1	20.0
61	226[9]	226	16	10	6	0.625	27.7
62	235[9]	235	16	16	10	1	38.8
63	207[9]	207	8	6	2	0.75	55.0
64	208[9]	208	4	2	2	0.5	78.6

[1]values shown for $L < 61$ are known to be optimal.
[2]runtime required to execute each experiment on a 266 MHz workstation under Linux.

Figure 3: Termination criterion 'B' results with KL.

4. LABS SOLVER COMPARISONS

Results with the KL solver under the termination criterion 'B', summarized in Figure 3, show significant improvement over earlier published results, maintaining quality of reported minima while reducing the cost of computation. However, the current performance still would not support similar evaluations for L much greater than 72. For $L = 77$, the *projected runtime* (using $1.552\text{e-}8*1.419^L$) is 0.94 years; if the processor performance would increase tenfold, it would still take 1.1 years to complete similar runs for $L = 84$. What is needed is a LABS solver with *both* (1) runtime complexity significantly better than $O(1.419^L)$ *and* (2) constant factor significantly better than 1.552e-8 relative to our 266 MHz platform.

We argue that the performance of any LABS solver under termination criterion 'B' can be assessed by its performance under the termination criterion 'A' as outlined in Figure 2. In this section we apply identical evaluation criteria to two solvers: KL solver (described in Section 2), and an ES solver which is based on the evolutionary strategies algorithm [7].

Evolutionary strategies (ES) in its standard form start with an initial parent population of λ sequences generated at random (here $\lambda = 10$). Then μ children are generated, each by selecting a parent at random and applying a randomly chosen mutation operation to it. From the μ children (here $\mu = 3 * L$), one selects the λ best individuals, which form the next parent population. The process is repeated until a termination criterion is met (either a known optimum is found – method 'A', or the process is repeated for an estimated number of generations – Method 'B'). The efficiency of an ES depends significantly on the type of mutations. Militzer et al. [7] developed a novel mutation operator that appears to work particularly well for the LABS problem.

Similar to KL, the ES solver also samples the LABS cost function, but now in a sequence of well-defined *generations* as outlined above. Our experiments demonstrate that the solvability function recorded for the ES solver, also in terms of *samples*, has an underlying exponential distribution similar to the one depicted for the KL solver in Figure 2. In fact, for the value of $L = 34$, the mean number of samples reported by the ES solver is 2,925,765 – slightly better than the mean number of samples 3,088,427 reported by the KL solver. A t-test declares both means to be equivalent, and a χ^2-test declares both distributions to be equivalent.

Comparison with a single value of L is not sufficient; we rely on the asymptotic analysis for averages and ratios of averages based on 128 experiments for consecutive values of $L = 20$ to $L = 47$ for both KL and ES solvers. As shown in Figure 4, we pair data reported by the two solvers in terms of average *samples*, average *runtime* (in seconds), and the *sampling efficiency* defined as the ratio of the average number of *samples* to the average *runtime*. For each set of data we also show a regression line, similar to the one for *restarts* in Figure 3, that can be used to *nominally* predict the asymptotic behavior of each solver:[2]

$$ES\_samples\ predictor \quad = \quad 5.558\text{E}{+}1 \times 1.370^{L} \qquad (4)$$

$$ES\_runtime\ predictor \quad = \quad 7.108\text{E}{-}4 \times 1.397^{L} \qquad (5)$$

$$ES\_efficiency\ predictor \quad = \quad 7.819\text{E}{+}4 \times 0.981^{L} \qquad (6)$$

$$KL\_samples\ predictor \quad = \quad 1.553\text{E}{+}1 \times 1.423^{L} \qquad (7)$$

$$KL\_runtime\ predictor \quad = \quad 1.287\text{E}{-}5 \times 1.463^{L} \qquad (8)$$

$$KL\_efficiency\ predictor \quad = \quad 1.206\text{E}{+}6 \times 0.973^{L} \qquad (9)$$

While we have not included the lower and upper bound predictors along with the nominal predictors, it is important to consider them in predictions of each solver's parameters *as well as* in comparisons between the two solvers. The upper and lower bounds on the predictor variability of the KL-solver track very well with the ones discussed in detail for *KL-restarts* in the preceding section: ratios of predicted to observed values can exceed a factor of 4.0. Similarly to the bounds shown for *KL-restarts* and including statistical variations, the bounds based on multiplying or dividing the nominal values by 5 are valid for both *KL_samples predictor* and *KL_runtime predictor*. Note however, that the variability of *KL_efficiency predictor* is negligible.

In contrast, data points reported by the ES-solver show relatively less variability, up to a factor of 2.0 above and 1.7 below the nominal *ES_samples predictor* – until we reach the value of $L = 43$. At this value, we record a nearly 14-fold increase above the nominal *ES_samples predictor*. Also, there is a reduction by a factor of 4.3 at $L = 46$. Clearly, addi-

---

[2]Only the *consecutive* values of $L = 20$ to $L = 47$ are used in the least-square fit to each data set. Data points for the ES-solver for $L = 50, 54, 58$ are shown for illustration only.
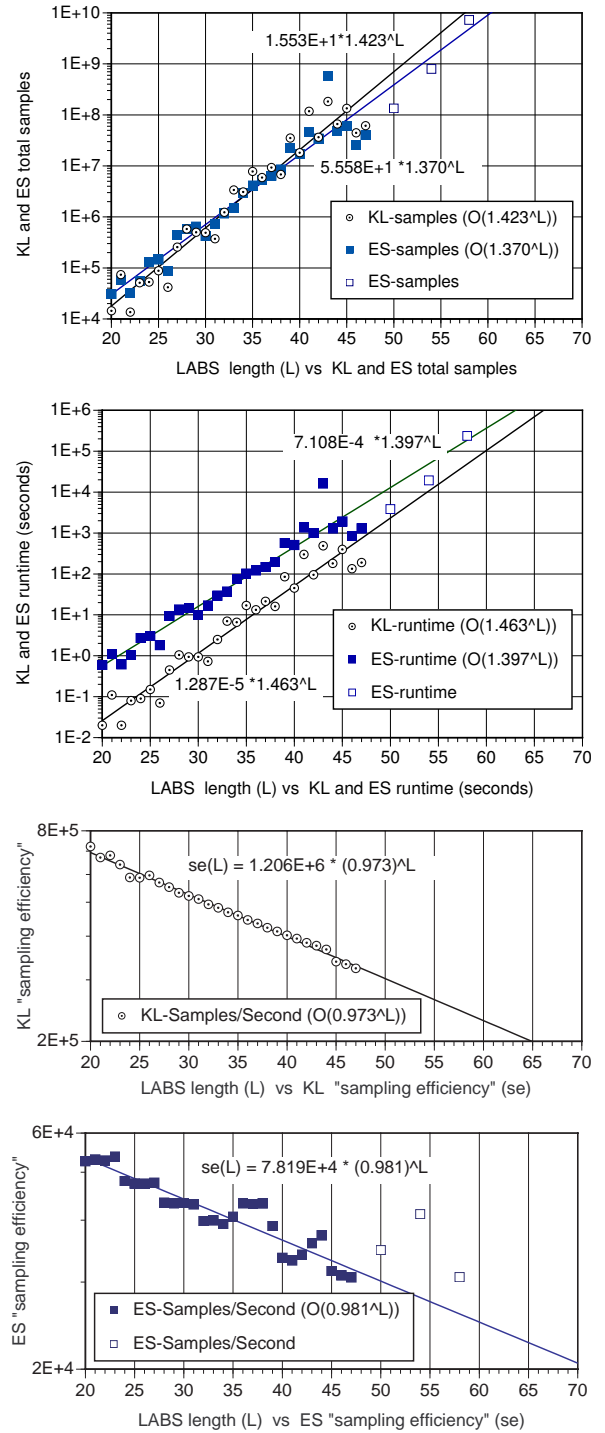

Figure 4: **Asymptotic comparisons of ES and KL.**

tioanl consecutive data points are needed to more completely characterize the asymptotic performance of the ES-solver; its performance variability may or may not exceed the variability demonstrated for the KL-solver up to the value of $L = 64$ in the preceding section. The ES-solver data points shown for values of $L = 50, 54, 58$ do track the respective regression lines closely for both *ES_samples predictor* and *ES_runtime predictor* – however, we have no information about the points in-between. Note also that the sampling efficiency of the ES-solver has significantly more variability,

starting at $L = 23$, than the one shown for the KL-solver.

Asympototically, the nominal performance of the ES-solver appears better: there is a crossover of the *samples* predictors at $L = 34$ and the *runtime* predictors at $L = 88$. However, these crossovers are not well-defined due to the demonstrated variability and most importantly, there is a significant difference in the *sampling efficiency* between the two solvers; for $L = 34$ the average *runtimes* are 6.60 and 74.5 seconds for KL and ES, respectively. A significant improvement of the sampling efficiency of the ES solver is needed to make it competitive, for $L < 88$, with the current version of the KL solver.

5. CONCLUSIONS AND FUTURE WORK

We have demonstrated a methodology that, while not guaranteeing optimum solutions for LABS problem, finds optima with high degree of confidence. For values of $L < 88$, the solver of choice is KL (at present). Our methodology, to compare two LABS solvers not only in terms of *runtime* but also *samples* required to find optimal solutions, provides new insights about the effectiveness and the efficiency of evolutionary strategies. Compared to KL, the ES-based search for optima may require significantly less samples (on average) as the problem size increases, however the challenge to reduce the cost of computing these samples should be addressed next by the ES-based search algorithms. The challenge for KL-based algorithms is clearly to reduce the number of samples required to find optimal solutions.

Reader is invited to visit the *LABSbed* home page [17], not only to access complete archives of experimental data summarized in this paper but also to access updates on (1) on LABS minima for values of $L > 64$ and a call for collaboration, (2) improved performance of new LABS solvers, (3) status of LABSbed tutorial, documentation, and ready-to-install solvers. A unique feature, intrinsic to the approach in this paper is the generation of multiple solutions for a given value of L. The number of these solutions in the table below is exact for values of $L \leq 35$. For values of $L > 35$,

L	0	1	2	3	4	5	6	7	8	9
				4	8	4	28	4	16	24
10	40	4	16	4	72	8	32	44	16	8
20	8	4	24	24	8	8	24	4	8	8
30	16	8	8	8	8	8	8	8	8	8
40	8	4	8	4	8	4	24	28	8	12
50	12	4	8	8	8	8	8	4	8	12
60	12	8	12	4	4					

the values listed are conjectured from our experiments to date. A total of 400 experiments under termination 'A' have been run for $L <= 47$, so values listed are most likely exact. Values for $L > 47$ are based on smaller number of experiments as listed in the table of Figure 3. Since it is clear that the number of solutions must always be a multiple of 4, we have rounded the solutions found to the nearest such value. For example, when we report the number of solutions for $L = 60$ as 10 in Figure 3, we report 12 in the table above. We anticipate new insights by (a) correlating the number of solutions to the observed and predicted variability in solver performance and (b) analyzing solution structures to improve the search process for the next generation of LABS solvers.

6. REFERENCES

[1] M. J. E. Golay. The merit factor of long low autocorrelation binary sequences. *IEEE: Trans. on Information Theory*, 28:543–549, 1982.

[2] J. Bernasconi. Low autocorrelation binary sequences: statistical mechanics and configuration space analysis. *J. Phys.*, 48:559–567, April 1987.

[3] S. Mertens. Exhaustive search for low-autocorrelation binary sequences. *Journal of Physics A: Mathematical and General*, 29:473–481, 1996.

[4] S. Mertens. Ground states of the Bernasconi model with open boundary conditions, 2003. See http://odysseus.nat.uni-magdeburg.de/~mertens/-bernasconi/open.dat.

[5] C. de Groot, D. Wurtz, and K. H. Hoffmann. Low autocorrelation binary sequences: Exact enumeration and optimization by evolutionary strategies. *Optimization (UK)*, 23(4):369–384, 1993.

[6] F.-M. Dittes. Optimization on rugged landscape: A new general purpose monte carlo approach. *Physical Review Letters*, 76:4651–4655, 1996.

[7] B. Militzer, M. Zamparelli, and D. Beule. Evolutionary search for low autocorrelated binary sequences. *IEEE Trans. on Evolutionary Comp.*, 2(1):34–39, April 1998.

[8] S. Prestwich. A Hybrid Search Architecture Applied to Hard Random 3-SAT and Low-Autocorrelation Binary Sequences. *The Sixth Int. Conf. on Principles and Practice of Constraint Programming, LNCS, Springer-Verlag*, 1894:337–352, 2000.

[9] J. Knauer. Home Page of LABS Problem Merit Factor Records, 2003. See http://www.cecm.sfu.ca/~jknauer/labs/records.html.

[10] F. Brglez, X. Y. Li, and M. F. Stallmann. The LABS Problem and an Encapsulation of Local Search Algorithms for Improved Performance and Reliability. *Technical Report*, 2003-TR@CBL-LABS, March 2003. Available at http://www.cbl.ncsu.edu/publications/.

[11] Woei-Kae Chen and Matthias F.M. Stallmann. Local search variants for hypercube embedding. In *Proc. 5th Distrib. Memory Computing Conf.*, pages 1375 – 1383, 1990.

[12] W.-K. Chen, M. Stallmann, and E.F. Gehringer. Hypercube embedding heuristics: An evaluation. *International Journal on Parallel Programming*, 18(6):505 – 549, 1989.

[13] B.W.Kernighan and S.Lin. An efficient heuristic procedure for partitioning graphs. *BSTJ*, pages 291–307, 1970.

[14] Holger H. Hoos and Thomas Stützle. Local Search Algorithms for SAT: An Empirical Evaluation. *Journal Of Automated Reasoning*, 24, 2000.

[15] F. Brglez, X. Y. Li, and M. Stallmann. On SAT Instance Classes and a Method for Reliable Performance Experiments with SAT Solvers. *Annals of Mathematics and Artificial Intelligence (AMAI), Special Issue on Satisfiability Testing*, 2003. Under review as the revision of the 2002-SAT Symposium paper. Available at http://www.cbl.ncsu.edu/publications/ .

[16] F. Brglez, M. F. Stallmann, and X. Y. Li. SATbed: An Environment For Reliable Performance Experiments with SAT Instance Classes and Algorithms. In *Proc. of SAT 2003, Sixth Int. Symp. on the Theory and Applications of Satisfiability Testing, May 5-8 2003, S. Margherita Ligure - Portofino, Italy*, 2003. A revised paper is available at www.cbl.ncsu.edu/publications/ .

[17] F. Brglez, M. Stallmann, and X. Y. Li. LABSbed Home Page: A Tutorial, A User Guide, A Software Archive, Archives of LABS Instances and Experimental Results, 2003. See www.cbl.ncsu.edu/OpenExperiments/LABS/ .