Exploiting Fitness Distance Correlation of Set Covering Problems

Markus Finger[1], Thomas Stützle[1], and Helena Lourenço[2]

[1] Darmstadt University of Technology, Intellectics Group,
Alexanderstr. 10, 64283 Darmstadt, Germany
[2] Universitat Pompeu Fabra, Deptartment of Economics and Business,
R. Trias Fargas 25-27, 08005 Barcelona, Spain

Abstract. The set covering problem is an $\mathcal{NP}$-hard combinatorial optimization problem that arises in applications ranging from crew scheduling in airlines to driver scheduling in public mass transport. In this paper we analyze search space characteristics of a widely used set of benchmark instances through an analysis of the fitness-distance correlation. This analysis shows that there exist several classes of set covering instances that have a largely different behavior. For instances with high fitness distance correlation, we propose new ways of generating core problems and analyze the performance of algorithms exploiting these core problems.

1 Introduction

The set covering problem (SCP) is a well-known problem in the area of Combinatorial Optimization. Its importance lies in the large number of real applications ranging from crew scheduling in airlines [13], driver scheduling in public transportation [15], and scheduling and production planning in several industries [20].

The SCP consists in finding a subset of columns of a zero-one $m \times n$ matrix such that it covers all the rows of the matrix at minimum cost. Let $M = \{1, 2, .., m\}$ and $N = \{1, 2, .., n\}$ be, respectively, the set of rows and the set of columns. Let $A = (a_{ij})$ be a zero-one matrix and $c = (c_j)$ be a n-dimensional integer vector that represents the cost of column j. We say that a column j covers a row i if $a_{ij} = 1$. The problem can be formally stated as

$$\begin{array}{ll} \text{Minimize} & v(SCP) = \sum_{i=1}^{n} c_i x_i \\ \text{s.t.} & \sum_{i=1}^{m} a_{ij} x_i \geq 1, \;\; i = 1, ..., m, \\ & x_i \in \{0, 1\}, \end{array}$$

where $x_j = 1$ if the column is in the solution, and $x_j = 0$, otherwise.

The simplicity of the SCP makes this model an ideal one to be used to solve real combinatorial optimization problems. To apply the SCP it is only necessary to have elements (rows) that need to be covered (served, assigned) at minimum cost by subsets (columns) of these elements. The rows can represent trips or pieces of work in a crew scheduling problem, customers with demand in a vehicle routing or tasks in a production scheduling. The columns can also represent a variety of elements as pairing or duties in crew scheduling to feasible routes in a vehicle routing problem.

Because the problem is $\mathcal{NP}$-hard and the instance in practice tend to be very large, a large number of solution approaches were proposed. Among the most successful algorithms are approximate algorithms, we review some of them in the next

section. Despite the many algorithmic approaches no insights into the search space characteristics of the SCP for approximate algorithms has been obtained. Yet, such insights are valuable to (i) understand algorithm behavior as well as they may propose new ways of attacking a problem. In this article, we analyze some search space characteristics of the SCP by an analysis of the fitness distance correlation [14] and show that very strong differences among the search space characteristics exist between different types of instances. Based on our insights from the analysis, we propose new ways of generating core problems, that is a much smaller SCP containing a subset of the most likely columns to appear in optimal solutions. Experimental results with a known SA algorithm [7] for the SCP show that our way of generating core problems is viable and obtains competitive results to other codes.

The paper is organized as follows. The next section presents an overview of the most prominent algorithmic approaches to the SCP. Section 3 presents the details of the fitness-distance analysis for the SCP and Section 4 gives computational results we obtained with an algorithm based on a new definition of core problems. We conclude in Section 6 and indicate directions for future work.

2 Algorithms for the set covering problem

2.1 Exact algorithms

In the Operations Research literature, an extensive number of papers are dedicated to the SCP and many exact and approximate algorithms were proposed to solve the SCP. Exact methods can solve instances with up to a few hundred rows and few thousand columns. Fisher and Kedia [12] present a B&B algorithm that can solve instances with 200 rows and 2000 columns. Christofides and Paixão [11] presented a tree-search exact procedure based on state-space relaxation and dynamic programming that can solve problems up to 400 rows and 4000 columns. A comparison of other exact algorithms like those of Beasley [5], Balas and Carrera [1] and general purpose ILP solvers can be found in [8].

2.2 Approximate algorithms

Approximate algorithms play an important role in solving the SCP, given the limitations of exact methods and the large list of applications using large size SCP. The proposed heuristic method to solve the SCP are based on a variety of techniques ranging from simple greedy construction algorithms over Lagrangian-based heuristics to metaheuristic approaches like simulated annealing or genetic algorithms. Pure greedy construction heuristics typically perform rather poorly. Very popular for the SCP are Lagrangian-based heuristics, which compute weights of the columns related to the likelihood that a column is part of an optimal solution. Several such heuristics exist with the most efficient ones being the recently proposed algorithms by Ceria, Nobili and Sassano [10] and the CFT heuristic by Caprara, Fischetti and Toth [9]. Since a few years, the number of applications of metaheuristics to the SCP has increased and extremely high performance is obtained from the most recent approaches. So far, the best results are mainly due to a Genetic Algorithm by Chu and Beasley [4], Simulated Annealing algorithms, the most recent being by Brusco, Jacobs and Thompson [7], and the iterated local search [16] algorithms by Marchiori and Steenbeck [18] and by Yagiura, Kishida and Ibaraki [17]. These latter

class	m	n	density (%)	no. inst.
4	200	1000	2	10
5	200	2000	2	10
6	200	1000	5	5
A	300	3000	2	5
B	300	3000	5	5
C	400	4000	2	5
D	400	4000	5	5
E	500	5000	10	5
F	500	5000	20	5
G	1000	10000	2	5
H	1000	10000	5	5

Inst.	m	n	density (%)	range of weights
AA03	106	8661	4.05	91-3619
AA04	106	8002	4.05	91-3619
AA05	105	7432	4.05	91-3619
AA06	105	6951	4.11	91-3619
AA11	271	4413	2.53	35-2966
AA12	272	4208	2.52	35-2966
AA13	265	4025	2.60	35-2966
AA14	266	3868	2.50	35-2966
AA15	267	3701	2.58	35-2966
AA16	265	3558	2.63	35-2966
AA17	264	3425	2.61	35-2966
AA18	271	3314	2.55	35-2966
AA19	263	3202	2.63	35-2966
AA20	269	3095	2.58	35-2966

Table 1. Details are given of benchmark instances from ORLIB (left table) and from Balas and Carrera (right table). Given are the number of rows (m), the number of columns (n), the density and for the AA* instance the range of weights.

two algorithms iteratively move through construction/destruction phases and local search phases.

One technique used in some of the best performing algorithms is that of generating core problems: in this case the algorithm runs on a subset of all columns that is likely to contain those occuring also in optimal solutions. Yet, core problems are often generated in a very ad-hoc manner.

3 Benchmark problems

For the fitness distance analysis we focused on two sets of benchmark instances. The first set, available from ORLIB at `http://mscmga.ms.ic.ac.uk/info.html` is composed of randomly generated instances with varying density and size, originally from [2] and [3]. This set comprises 65 instances (see Table 1). In these instances the column costs are uniformly distributed in the interval $[1, 100]$, each column covers at least one row, and each row is covered by at least two columns.

The second set with instances aa03-aa06 and aa11-aa20 stems from the paper of Balas and Carrera [1] (see Table 1, right side). There are 14 instances of different size and different density; additionally, the instances differ in the range of the column weights.

4 Fitness Distance Analysis

The performance of metaheuristics depends strongly on the shape of the underlying search space. Central to the search space analysis of combinatorial optimization problems is the notion of *fitness landscape* [19, 21]. Intuitively, the fitness landscape can be imagined as a mountainous region with hills, craters, and valleys. The performance of metaheuristics strongly depends on the ruggedness of the landscape, the distribution of the valleys, craters and the local minima in the search space, and the overall number of the local minima.

Formally, the fitness landscape is defined by (i) the set of all possible solutions $\mathcal{S}$, (ii) an objective function that assigns to every $s \in \mathcal{S}$ a fitness value $f(s)$, and (iii) a distance measure $d(s, s')$ which gives the distance between solutions s and s'. The fitness landscape determines the shape of the search space as encountered by a local search algorithm.

For the investigation of the suitability of a fitness landscape for adaptive multi-start algorithms like the best performing metaheuristics for the SCP [7, 18, 17], the analysis of the correlation between solution costs and the distance between solutions or to globally optimal solutions has proved to be a useful tool [6, 14]. The fitness distance correlation (FDC) [14] measures the correlation of the solution cost and the distance to the closest global optimum. Given a set of cost values $C = \{c_1, \ldots, c_m\}$ and the corresponding distances $D = \{d_1, \ldots, d_m\}$ to the closest global optimum the correlation coefficient is defined as:

$$r(C, D) = \frac{c_{CD}}{s_C \cdot s_D}, \text{where } c_{CD} = \frac{1}{m} \sum_{i=1}^{m} (c_i - \bar{c})(d_i - \bar{d})$$

and $\bar{c}$, $\bar{d}$ are the average cost and the average distance, s_C and s_D are the standard deviations of the costs and distances, respectively.

One difficulty for applying the FDC analysis to the SCP is that straight-forward distance measure exists, because of the SCP being a sub-set problem. Therefore, we rather use the closeness between solutions (based on the closeness, also a measure for the distance between solutions may be defined):

Definition 1. *Let $s, s' \in \mathcal{S}$ be two feasible solutions for an SCP, then we define the closeness $n(\cdot, \cdot)$ and the distance $d(\cdot, \cdot)$ between s and s' as*

$$n(s, s') = \text{number of same columns in } s \text{ and } s\text{'}$$
$$d(s, s') = \max(|s|, |s'|) - n(s, s')$$

A maximal distance $d(s, s') = \max(|s|, |s'|)$ means that both solutions do not have any column in common, if $d(s, s') = 0$ we have $s = s'$.

As a first step, we generated a number of best-known solutions for all the instances under concern using a variant of the SA of Brusco, Jacobs, and Thompson [7]. For instances where optimal solutions are not available, it is conjectured that the best-known solutions are actually optimal. In a second step, we generated 1000 locally optimal solutions, starting from random initial solutions, using the local search algorithm given in Figure 1. The function $\mathrm{SN}_1(s, j)$ performs the following steps: (i) it removes the column j from the current solution s, (ii) it completes s by iteratively choosing randomly a still uncovered column and adding a column with best value of $c_i/cv(i)$, where $cv(i)$ is the cover value of column i, that is, the number of still uncovered rows that is covered column i, and (iii) it removes redundant columns.

4.1 Results on ORLIB instances

Results on the FDC analysis are plotted in Figure 2[1] and detailed results are available in Table 4.1. In general, on the ORLIB instances we observed a high, positive

[1] In these plots the points are stratified according to the solution quality. The reason is that only few different values are possible for the solution quality. For example, the

```
Procedure IV^f_{N_1};
    S := KH_1(∅);
    while ( improvement ) do
        R := S;
        while ( R ≠ ∅ ∧ no improvement found so far ) do
            choose randomly a column j ∈ R and R := R \ {j};
            s' := SN_1(s, j);
            if (f(s') < f(s)) then s := s';
        end while;
    end while;
    return S;
end IV^f_{N_1};
```

Fig. 1. First-improvement local search procedure $IV^f_{\mathcal{N}_1}$.

correlation between the solution quality and the closeness to optimal solutions for all instances.

The high values for the correlation coefficient (see Table 4.1) confirm the observation from the plots in Figure 2. Only for one instance (C.3) is the correlation coefficient below 0.25. The solution quality of the local optima is relatively high and for some of the instances, the best local optima among the 1000 generated, even matched the best known solutions. Particularly interesting are two observations. First, the average percentage closeness between the local optima and the total number of all different columns in the 1000 local optima depends strongly on the size of the instance as well as their densities: the larger the density the smaller is the average closeness among local optima, the less columns are in local optima and the less is the number of distinct columns in the 1000 local optima. Second, there is a direct relationship between problem size and local optima statistics: The larger the instance, the more distinct columns are encountered in the local optima (compare instance classes C and G and instance classes D and H–they have the same densities). Interestingly, for some instance classes (with same density) the average closeness of the local optima actually increases, like it is the case for C and G. This may indicate that these problem do not really become harder with instance size.

4.2 Results on instances from Balas and Carrera

The FDC plots of some instances from Balas and Carrera (BC) are given in Figure 3, detailed results are given in Table 4.2. The search space analysis of these instances shows a very different behavior. First, the closeness of the instances to the best known solutions is very low; this can be seen when inspecting the plots: all points are in the range of a closeness between 0-35%; whereas for the ORLIB instances the closeness was mostly larger than 0.5. Second, there is only a very small correlation between fitness and closeness to the best known solutions, showing that the objective function value provides much less guidance towards the global optima than on the ORLIB instances. Third, the average closeness of the local optima is very low,

instance E4 has an optimal solution of 28 and therefore, a solution of 29 has a deviation of around 3.5% from the optimal solution.

PI	best known sol.	r	no. LO	sol. quality local minima			$\overline{C(LO)}$ (in %)	$\sum$ col (LO)	GO
				avg.	best	worst			
C.1	227	0.5955	762	236.67	229	272	74.58	252	1
C.2	219	0.5741	964	234.76	221	262	65.66	292	11
C.3	243	0.1355	989	266.55	252	296	64.07	310	1
C.4	219	0.6400	945	237.70	225	268	64.02	290	113
C.5	215	0.7409	870	224.48	215	270	71.02	257	7
D.1	60	0.7238	776	64.31	60	79	29.75	152	9
D.2	66	0.6306	864	70.93	66	82	31.68	166	42
D.3	72	0.4653	929	78.18	73	89	31.81	191	25
D.4	62	0.5522	664	66.73	62	79	35.61	160	3
D.5	61	0.7868	297	65.74	61	76	35.22	148	2
E.1	29	0.7202	836	30.90	29	36	17.65	119	163
E.2	30	0.2944	992	33.33	30	40	13.54	151	1
E.3	27	0.4699	962	29.92	27	35	14.39	132	1
E.4	28	0.5516	961	30.68	28	35	16.44	123	6
E.5	28	0.7602	844	30.08	28	38	17.61	126	38
F.1	14	0.6475	979	15.51	14	18	5.78	91	55
F.2	15	0.7462	868	16.26	15	19	7.22	90	98
F.3	14	0.4428	925	16.24	14	19	6.53	84	1
F.4	14	0.5802	991	15.68	14	18	5.42	90	28
F.5	13	0.2661	997	15.17	14	17	4.44	88	1
G.1	176	0.7667	995	189.31	178	204	85.93	421	66
G.2	154	0.5016	1000	165.96	158	183	82.42	404	2
G.3	166	0.4355	999	177.11	170	192	90.60	396	5
G.4	168	0.3989	1000	181.18	173	201	82.26	422	1
G.5	168	0.4897	999	181.68	173	198	81.39	439	33
H.1	63	0.3646	1000	69.57	64	77	33.11	297	1
H.2	63	0.5301	1000	69.34	64	77	32.79	298	2
H.3	60	0.4107	1000	65.90	61	77	31.57	275	1
H.4	58	0.5582	1000	63.66	59	69	31.02	277	45
H.5	55	0.6344	1000	60.25	56	66	33.58	258	45

Table 2. Results of the FDC analysis of ORLIB instances. Given are the instance identifier (PI), the best known solutions, the correlation coefficient r, the number of distinct local optima, the best, average, and worst solution found, the average closeness, the number of different columns in the 1000 local optima, and the number of best known solutions (GO) used in the FDC analysis.

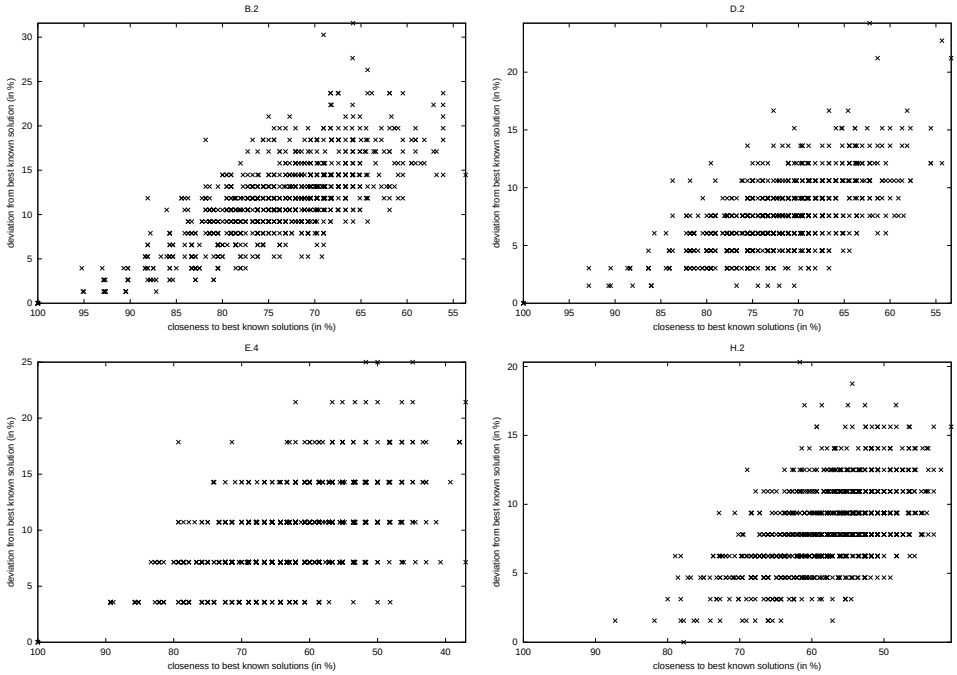

Fig. 2. Fitness distance graphs for some selected ORLIB instances.

leading to the fact that the overall number of distinct columns in the local optima is extremely high.

These results suggest that the BC instances are much harder to solve for local search than the ORLIB instances because less guidance is given by the objective function and a much larger number of columns actually appear in good solutions, leading to a larger *effective* search space.

5 Core problems

The results of the search space analysis can be exploited to find systematic and well justified ways for reducing instance size by defining small core problems: generate some local optima according to the algorithm of Figure 1 and define the core problem as the union of the columns contained in the local optima. Such an approach is interesting in a situation like observed for the ORLIB problems, where different local optima show a similar structure and the structure of local optima correlates highly with that of global optima. In such a case, the columns of the global optimum also are likely to occur in many of the local optima with a high frequency.

For the generation of core problems two issues are important: First, a sufficient number of different local optima has to be generated. Second, the overall time of generating core problems and then solving an instance should be smaller than solving an instance without using core problems.

We tested this idea using a variant of the Simulated Annealing algorithm by Brusco et al. [7] on the ORLIB instances which is run on the initial problem (SA) and

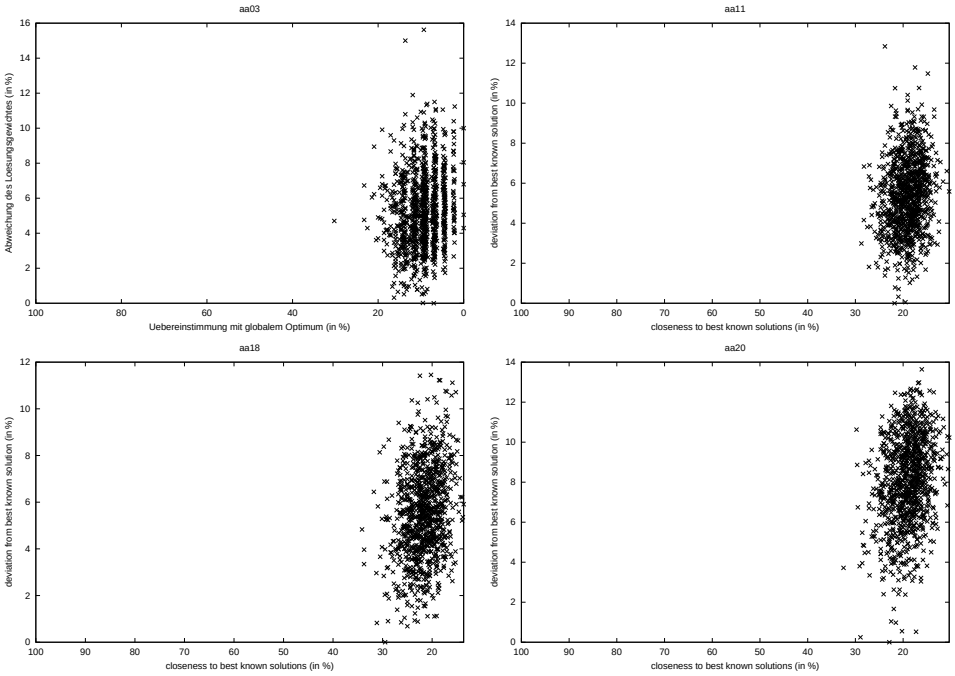

Fig. 3. Fitness distance graphs for some selected instances of Balas and Carrera.

on the core problems SA_{core}. For the generation of the core problems, we construct 25 starting solutions for the local search 1 by a randomized greedy construction heuristic and the core problem then contains all columns member of any of the 25 local optima. The greedy construction heuristic iteratively selects first an still uncovered row and in a second step it chooses randomly among the five highest ranked columns, where the rank of a column is determined by its cost divided by the cover value. (The use of a greedy algorithm has the advantage that the subsequent local search is much faster than from random starting solutions.)

The results of SA_{core} are in Table 5, the results of SA together with a tabu search variant ($\mathrm{IVT}^b_{\mathcal{N}_1}$) and a zero temperature SA algorithm (SA_{T_0}) are in Table 4. (Let us remark here that our results (by SA or SA_{core}) to a large extent match the results obtained with the best performing algorithms for the SCP; only on some few problems slightly better average solution qualities have been reported in [7, 9, 18, 17].) All results in Tables 4 and 5 are based on 25 independent trials; in the case of the results on the core problems, for each trial a new core problem was generated; all algorithms were limited to 2000 iterations. The programs were coded with C++ and run on a Pentium III 700MHz CPU.

The computational results show that the SA working on core problems is significantly faster, which is most visible for the largest instances of the class H. The speed-up only comes with a very minor loss in solution quality on some few instances when compared to the SA working on all columns; for some few instances even better performance could be obtained. We expect that on the few instances, where SA_{core} did not match the best-known solutions, this could be achieved by

PI	best known sol.	r	no. LO	sol. quality local minima			$\overline{C(LO)}$ (in %)	$\sum$ col (LO)	GO
				avg.	best	worst			
aa03	33155	0.1486	1000	36319.21	34485	39873	2.89	3565	257
aa04	34573	0.2072	1000	37650.96	34731	40803	3.89	3057	704
aa05	31623	0.0071	1000	33027.26	32235	36237	4.43	2915	155
aa06	37464	0.1262	1000	40701.95	38762	44345	4.47	2741	458
aa11	35384	0.1702	1000	39843.31	37804	42658	15.00	3733	64
aa12	30809	0.1936	1000	35385.19	32962	38114	13.82	3576	1202
aa13	33211	0.2092	1000	38143.74	36260	40836	12.04	3424	339
aa14	33219	0.2114	1000	37209.92	35441	40318	13.67	3374	48
aa15	34409	0.1943	1000	38950.44	36903	41390	14.65	3043	128
aa16	32752	0.1865	1000	37983.32	35302	40494	13.86	3137	271
aa17	31612	0.2480	1000	36608.98	34487	39334	14.26	2866	400
aa18	36782	0.2407	1000	41182.28	39013	43482	16.96	2806	1024
aa19	32317	0.2464	1000	36893.85	34395	39345	10.60	2774	4356
aa20	34912	0.2583	1000	40114.32	37116	42179	15.79	2663	55

Table 3. Results of the FDC analysis of ORLIB instances. For a description of the entries see Table 3.

increasing the number of local optima for generating core problems. It is interesting to see that the largest part of the computation time actually is spent on the generation of the core problems and not by the SA. The main reason is certainly that the number of columns in the core problems is very small (see last column of Table 5).

6 Conclusions

We have analyzed some search space characteristics of the SCP and, based in this analysis we proposed new, systematic ways of generating core problems for the SCP. Computational results on the SCP instances from ORLIB showed that the resulting core problems are very small, but for the ORLIB instances they often contain all columns necessary to find the best-known solutions. In fact, some of the core problems were solved using an exact algorithm and we could verify, that the best solutions found by an Simulated Annealing algorithm working on the core problems, identified consistently the optimal solutions with respect to the core problems.

There are several ways how this work can be extended. First, we could extend our analysis to other instances from real applications. Second, faster ways of identifying core problems would be interesting, because this task consumes the largest part of the computation time of SA_{core}. Third, our approach could be enhanced by dynamically changing the core problems during the run of the algorithm as done in [9, 18]. Fourth, the same ideas of generating core problems could also be applied to other problems that show significant fitness distance correlation. The results on the SCP suggest that this last idea is very promising for a number of combinatorial problems.

References

1. E. Balas and M.C. Carrera. A dynamic subgradient-based branch and bound procedure for set covering. *Operations Research*, 44:875–890, 1996.

PI	best known sol.	$IVT^b_{\mathcal{N}_1}$			SA			SA_{T_0}		
		average LG	no. opt	time	average LG	no opt	time	average LG	no opt	time
E.1	29	29	25	11.69	29	25	1.60	29	25	0.53
E.2	30	30.84	4	147.42	30	25	0.26	30	25	0.24
E.3	27	27.08	23	325.86	27	25	0.22	27	25	0.21
E.4	28	28	25	249.21	28	25	7.47	28	25	8.06
E.5	28	28	25	50.94	28	25	2.07	28	25	1.35
F.1	14	14	25	120.61	14	25	4.85	14	25	6.36
F.2	15	15	25	19.03	15	25	0.50	15	25	1.78
F.3	14	14	25	421.88	14	25	39.17	14	25	64.40
F.4	14	14	25	212.54	14	25	12.49	14	25	17.95
F.5	13	13.88	3	65.65	13.24	19	77.80	13.92	2	14.35
G.1	176	177.16	8	402.16	176	25	16.49	177.32	11	23.78
G.2	154	154.56	13	425.52	155	0	28.25	156.72	0	10.48
G.3	166	167.52	0	290.98	167.32	2	38.90	167.6	0	28.92
G.4	168	169.8	5	325.94	168.68	15	43.81	170.88	3	29.35
G.5	168	169	11	372.57	168	25	18.64	168.8	14	25.62
H.1	63	64.12	0	756.83	63.92	2	126.11	64.2	0	116.98
H.2	63	64.24	19	631.37	63.16	21	179.74	63.96	2	73.76
H.3	59	60.64	9	580.45	59.28	19	255.89	60.28	18	177.98
H.4	58	59.04	6	498.56	58	25	198.00	58.6	11	164.52
H.5	55	55.36	16	425.97	55	25	35.92	55.28	20	67.30

Table 4. Computational results with Tabu Search, Simulated Annealing, and zero temperature annealing.

2. J.E. Beasley. An algorithm for the set covering problems. *European Journal of Operational Research*, 31:85–93, 1987.
3. J.E. Beasley. A lagrangian heuristic for set covering problems. *Naval Research Logistics*, 37:151–164, 1990.
4. J.E. Beasley and P.C. Chu. A genetic algorithm for the set covering problem. *European Journal of Operational Research*, 94:392–404, 1996.
5. J.E. Beasley and K. Jornsten. Enhancing an algorithm for set covering problems. *European Journal of Operational Research*, 58:293–300, 1992.
6. K.D. Boese, A.B. Kahng, and S. Muddu. A New Adaptive Multi-Start Technique for Combinatorial Global Optimization. *Operations Research Letters*, 16:101–113, 1994.
7. M.J. Brusco, L.W. Jacobs, and G.M. Thompson. A morphing procedure to supplement a simulated annealing heuristic for cost- and coverage-corrleated set covering problems. *Annals of Operations Research*, 86:611–627, 1999.
8. A. Caprara, M. Fischetti, and P. Toth. Algorithms for the set covering problem. Technical Report OR-98-3, DEIS, University of Bologna, Italy, 1998.
9. A. Caprara, M. Fischetti, and P. Toth. A heuristic method for the set covering problem. *Operations Research*, 47:730–743, 1999.
10. S. Ceria, P. Nobili, and A. Sassano. A lagrangian-based heuristic for large-scale set covering problems. *Mathematical Programming*, 81:215–228, 1998.
11. N. Christofides and J.P. Paix ao. Algorithms for large scale set covering problems. *Annals of Operations Research*, 43:261–277, 1993.
12. M.L. Fisher and P. Kedia. Optimal solution of set covering/set partitioning problems using dual heuristics. *Management Science*, 36:674–688, 1990.

Problem	best known sol.	SA_{core}								
		average	solution quality			time	no.	$time_{LO}$	Core-problem	
		sol.qual	best	no. opt.	worst	(sec)	LO		m	No. columns
E.1	29	29	29	25	29	0.008	25	2.315	500	83.2
E.2	30	31	31	25	31	0.073	25	3.155	500	91.72
E.3	27	27.04	27	24	28	0.000	25	2.616	500	86.16
E.4	28	28	28	25	28	0.101	25	2.035	500	85.4
E.5	28	28	28	25	28	0.055	25	2.508	500	83.96
F.1	14	14	14	25	14	0.020	25	1.961	500	53.08
F.2	15	15	15	25	15	0.006	25	2.184	500	49.28
F.3	14	14	14	25	14	0.110	25	2.622	500	51.4
F.4	14	14	14	25	14	0.081	25	1.645	500	49.12
F.5	13	14	14	25	14	0.000	25	1.450	500	47.56
G.1	176	176	176	25	176	3.152	25	20.246	1000	301.48
G.2	154	155.04	154	1	156	5.776	25	17.730	1000	279.24
G.3	166	167.52	166	3	169	16.414	25	17.202	1000	283.08
G.4	168	168.48	168	19	170	11.646	25	19.640	1000	286.16
G.5	168	168.12	168	23	170	2.997	25	22.417	1000	296.32
H.1	63	64	64	25	64	2.663	25	31.347	1000	183
H.2	63	63.12	63	22	64	7.883	25	31.473	1000	179.36
H.3	59	59.56	59	15	61	6.015	25	27.839	1000	185.76
H.4	58	58.28	58	18	59	7.227	25	28.519	1000	179
H.5	55	55	55	25	55	0.537	25	28.436	1000	175.6

Table 5. Computational results with Simulated Annealing using core problems.

13. E. Housos and T. Elmoth. Automatic optimization of subproblems in scheduling airlines crews. *Interfaces*, 27(5):68–77, 1997.
14. T. Jones and S. Forrest. Fitness distance correlation as a measure of problem difficulty for genetic algorithms. In L.J. Eshelman, editor, *Proc. of the 6th Int. Conf. on Genetic Algorithms*, pages 184–192. Morgan Kaufman, 1995.
15. H. Lourenço, R. Portugal, and J.P. Paix ao. Multiobjective metaheuristics for the bus-driver scheduling problem. *Transportation Science*, 35(3):331–343, 2001.
16. H.R. Lourenço, O. Martin, and T. Stützle. Iterated local search. In F. Glover and G. Kochenberger, editors, *Handbook of Metaheuristics*. Kluwer Academic Publishers, Boston, MA, USA, 2002. to appear.
17. M. Kishida M. Yagiura and T. Ibaraki. A 3-flip neighborhood local search for the set covering problem. submitted for publication, 2001.
18. E. Marchiori and A. Steenbeek. An evolutionary algorithm for large scale set covering problems with application to airline crew scheduling. In *Real World Applications of Evolutionary Computing*, volume 1083 of *Lecture Notes in Computer Science*, pages 367–381. Springer Verlag, Berlin, Germany, 2000.
19. P.F. Stadler. Towards a theory of landscapes. Technical Report Technical Report SFI–95–03–030, Santa Fe Institute, 1995.
20. F.J. Vasko and F.E. Wolf. Optimal selection of ingot sizes via set covering. *Operations Research*, 15:115–121, 1988.
21. E.D. Weinberger. Correlated and uncorrelated fitness landscapes and how to tell the difference. *Biological Cybernetics*, 63:325–336, 1990.