Effective Variable Fixing and Scoring Strategies for Binary Quadratic Programming

Yang Wang[1], Zhipeng Lü[1], Fred Glover[2], and Jin-Kao Hao[1]

[1] LERIA, Université d'Angers, 2 Boulevard Lavoisier, 49045 Angers Cedex 01, France
[2] OptTek Systems, Inc., 2241 17th Street Boulder, CO 80302, USA
{yangw,lu,hao}@info.univ-angers.fr, glover@opttek.com

Abstract. We investigate two variable fixing strategies and two variable scoring strategies within a tabu search algorithm, using the unconstrained binary quadratic programming (UBQP) problem as a case study. In particular, we provide insights as to why one particular variable fixing and scoring strategy leads to better computational results than another one. For this purpose, we perform two investigations, the first analyzing deviations from the best known solution and the second analyzing the correlations between the fitness distances of high-quality solutions. We find that one of our strategies obtains the best solutions in the literature for all of the test problems examined.

1 Introduction

The strategy of fixing variables within optimization algorithms (also sometimes called *backbone* guided search) often proves useful for enhancing the performance of methods for solving constraint satisfaction and optimization problems [10, 11]. Such a strategy was proposed in early literature [1] as a means for exploiting critical variables identified as *strongly determined* and *consistent*, and has come to be one of the basic strategies associated with tabu search. Two of the most important features of this strategy are to decide how to score the variables (variable scoring) and which variables should be fixed (variable fixing).

In this paper, we provide a case study of variable fixing and scoring strategies within a tabu search variable fixing and scoring (TS/VFS) algorithm designed to solve the Unconstrained Binary Quadratic Programming problem

$$\text{UBQP: Maximize } f(x) = x'Qx \qquad x \text{ binary}$$

where Q is an n by n matrix of constants and x is an n-vector of binary variables.

The formulation UBQP is notable for its ability to represent a wide range of important problems, as noted in the survey of [6]. Motivated by this extensive range of applications, a number of advanced heuristic and metaheuristic algorithms have been devised for solving the UBQP problem ([2, 7–9]). However, to date there exist no studies of methods that employ variable fixing and scoring strategies within these algorithms. In this work, we investigate different variable fixing and scoring strategies within our TS/VFS algorithm and undertake to answer related questions such as: why does one particular variable fixing and scoring strategy lead to better computational results than another one? Which aspects are more important in designing effective optimization algorithms using variable fixing and scoring?

To this end, we present an experimental analysis of two variable fixing strategies and two variable scoring strategies within the TS/VFS algorithm. The analysis shows that the computational results strongly depend on the variable fixing strategies employed but are not very sensitive to the variable scoring methods. Moreover, the analysis sheds light on how different fixing and scoring strategies are related with the search behavior and the search space characteristics.

2 The TS Variable Fixing and Scoring Algorithm

Algorithm 1 describes the framework of our TS/VFS algorithm. It begins with a randomly constructed initial solution x^s and repeatedly alternates between a tabu search procedure and a phase that either fixes or frees variables until a stop criterion is satisfied. The TS procedure is employed for *maxIter* iterations to improve the input solution and to obtain p best solutions cached in P as the reference solutions, which are used to score and fix variables.

Algorithm 1 Pseudo-code of the TS/VFS algorithm for UBQP

```
 1: Input: matrix $Q$
 2: Output: the best binary $n$-vector $x^*$ found so far
 3: $x^* = \emptyset$; $f(x^*) = -\infty$; $f_p = -\infty$; $C = \emptyset$
 4: repeat
 5:    Construct an initial solution $x^s$ ($x^s_j = x^0_j, j \in C$)          (Section 3.2)
 6:    $x' \leftarrow$ TabuSearch($x^s$, $maxIter$)                             (Section 2.2)
 7:    Keep $p$ best solutions found during $TabuSearch$ in population $P$, $|P| = p$
 8:    if $f(x') > f(x^*)$ then
 9:       $x^* = x'$
10:    end if
11:    if $f(x') > f_p$ then
12:       $VarScore \leftarrow$ FixingScoringStrategy($P$)                      (Section 2.1 & 3.1)
13:       $VarSorted \leftarrow$ FixingScoreSorting($VarScore$)
14:       $FixedVar \leftarrow$ FixingStrategy($VarSorted$, $FixedNum$)         (Section 3.2)
15:    else
16:       $VarScore \leftarrow$ FreeingScoringStrategy($P$)                     (Section 2.1 & 3.1)
17:       $VarSorted \leftarrow$ FreeingScoreSorting($VarScore$)
18:       $FixedVar \leftarrow$ FreeingStrategy($VarSorted$, $DroppedNum$)      (Section 3.2)
19:    end if
20:    $f_p = f(x^*)$
21: until a stop criterion is satisfied
```

If the objective value $f(x')$ obtained by the current round of TS is better than the previous one f_p, a variable fixing phase is launched. Specifically, the fixing phase consists of three steps: *FixingScoringStrategy*, *FixingScoreSorting* and *FixingStrategy*. *FixingScoringStrategy* is used to give score values to variables and then *FixingScoreSorting* to sort the variables according to these values. *FixingStrategy* determines a number *FixedNum* of variables that go into a set *FixedVar* of variables to be fixed whose index set F is referenced in Algorithm 1. Consequently, the set of variables *FixedVar* will not be allowed to change its composition during the next round of TS, although conditionally changing the value of a fixed variable is another interesting strategy worthy of further investigation. It is understood that the values of variables x_j^s in the starting solution x^s are selected randomly except for $j \in F$.

On the contrary, if the TS procedure fails to find an improved solution relative to f_p, the algorithm performs the freeing phase to release some of the fixed variables to permit these variables to change their values during the next round of TS. Similar to the fixing phase, the freeing phase also consists of three steps. To describe these steps we make use of the following definitions.

2.1 Definitions

Definition 1. Relative to a given solution $x' = \{x'_1, x'_2, ..., x'_n\}$ and a variable x_i, the (objective function) *contribution* of x_i in relation to x' is defined as:

$$VC_i(x') = (1 - 2x_i)(q_{ii} + \sum_{j \in N \setminus \{i\}} q_{ij} x_j) \tag{1}$$

As noted in [2] and in a more general context in [4], $VC_i(x')$ identifies the change in $f(x)$ that results from changing the value of x'_i to 1 - x'_i; i.e.,

$$VC_i(x') = f(x'') - f(x') \tag{2}$$

where $x''_j = x'_j$ for $j \in N - \{i\}$ and $x''_i = 1 - x'_i$. We observe that under a maximization objective if x' is a locally optimal solution, as will typically be the case when we select x' to be a high quality solution, then $VC_i(x') \leq 0$ for all $i \in N$, and the current assignment $x_i = x'_i$ will be more strongly determined as $VC_i(x')$ is "more negative".

Definition 2. Relative to a given population of solutions $P = \{x^1, \ldots, x^p\}$ and their corresponding objective values $FV = \{f(x^1), \ldots, f(x^p)\}$ indexed by $I = \{1, \ldots, p\}$, and relative to a chosen variable x_i, let $P_i(0) = \{k \in I : x_i^k = 0\}$ and $P_i(1) = \{k \in I : x_i^k = 1\}$, the (objective function) *contribution* of x_i in relation to P is defined as follows.

Contribution for $x_i = 0$:

$$VC_i(P:0) = \sum_{k \in P_i(0)} (\beta \cdot VC_i(x^k) + (1 - \beta) \cdot \tilde{A}(f(x^k)) \cdot VC_i(x^k)) \tag{3}$$

Contribution for $x_i = 1$:

$$VC_i(P:1) = \sum_{k \in P_i(1)} (\beta \cdot VC_i(x^k) + (1 - \beta) \cdot \tilde{A}(f(x^k)) \cdot VC_i(x^k)) \tag{4}$$

where f_{min} and f_{max} are respectively the minimum and maximum objective values of the set FV and $\tilde{A}(\cdot)$ represents the normalized function:

$$\tilde{A}(f(x^k)) = (f(x^k) - f_{min})/(f_{max} - f_{min} + 1) \tag{5}$$

Notice that this scoring function not only considers the contributions of the variables but also the relative quality of the solution with respect to other reference solutions in the population P. Relative to variable x_i and a population P, The score of x_i is then defined as:

$$Score(i) = max\{VC_i(P:0), VC_i(P:1)\} \tag{6}$$

2.2 Tabu Search

Our TS procedure begins from a starting solution x^s as indicated in Algorithm 1 and uses a *neighborhood* defined by the simple *one-flip move*. Once the method is launched, the variables in $FixedVar$ are held fixed during the execution of the TS procedure. The method incorporates a *tabu list* as a "recency-based" memory structure to assure that solutions visited within a certain span of iterations, called the tabu tenure, will not be revisited [3]. In our implementation, we elected to set the tabu tenure by the assignment $TabuTenure(i) = tt + rand(10)$, where tt is a given constant $(n/100)$ and rand(10) takes a random value from 1 to 10. Interested readers are referred to [4, 7] for more details.

2.3 Reference Solutions

Reference solutions are used for fixing or freeing variables. We conjecture that there exists a subset of variables, of non-negligible size, whose optimal values are often also assigned to these same variables in high quality solutions. Thus, our goal is to identify such a critical set of variables and infer their optimal values from the assignments they receive in high quality solutions. Our expectation is that this will reduce the search space sufficiently to enable optimal values for the remaining variables to be found more readily. On the basis of this conjecture, we maintain a set of reference solutions consisting of good solutions obtained by TS. Specifically, we take a given number p of the best solutions from the current round of TS (subject to requiring that these solutions differ in a minimal way), which then constitute a solution population P for the purpose of fixing or freeing variables. In our implementation, we empirically set $p = 20$. (A more refined analysis is possible by a strategy of creating clusters of the solutions in the reference set and of considering interactions and clusterings among subsets of variables as suggested in [1].)

2.4 Variable Fixing Procedure

Given the reference solutions, our variable fixing procedure consists of three steps: *FixingScoringStrategy*, *FixingScoreSorting* and *FixingStrategy*. Variables are scored using Eq. (6) and sorted according to their scores in a non-decreasing order. We then decide how many variables with the smallest contribution scores should be fixed at their associated values x'_i at the current level and fix them so that these variables are compelled to receive their indicated values upon launching the next round of TS.

Let $Fix(h)$ denote the number of new variables (na) that are assigned fixed values and added to the fixed variables at fixing phase h. We begin with a chosen value $Fix1$ for $Fix(1)$, referring to the number of fixed variables at the first fixing phase and then generate values for higher fixing phases by making use of an "attenuation fraction" g as follows. We select the value $Fix1 = 0.25n$ and the fraction $g = 0.4$.

$$Fix(1) = Fix1$$
$$Fix(h) = Fix(h-1) \cdot g \text{ for } h > 1$$

2.5 Variable Freeing Procedure

Experiments demonstrate that in most cases, the fixed variables match well with the putative optimal solution. Nevertheless, it is possible that some of these variables are wrongly fixed, resulting in a loss of effectiveness of the algorithm. In order to cope with this problem, it is imperative to free the improperly fixed variables so that the search procedure can be put on the right track.

Like the fixing procedure, the freeing procedure also consists of three steps: *FreeingScoringStrategy*, *FreeingScoreSorting* and *FreeingStrategy*. Contrary to the fixing phase, the number of the variables freed from their assignments at each freeing phase is not adjusted, due to the fact that at each phase only a small number of variables are wrongly fixed and need to be freed. Specifically, we set the number nd of fixed variables to free to 60. Then, these selected fixed variables are free to receive new values when initiating the next round of TS.

3 Variable Scoring and Fixing Strategies

3.1 Variable Scoring Strategy

We introduce two variable scoring strategies: the first one only considers the contribution of the variables (Definition 1) in the reference solutions while the second one simultaneously considers this contribution and the quality of reference solutions. By Definition 2 in Section 2.1, the part of the equation multiplied by $1-\beta$ is obviously equal to 0 if β is 1.0, which implies that the objective values of the reference solutions are neglected. This constitutes our first variable scoring strategy. We introduce the second variable scoring strategy by simultaneously considering the solution quality of the reference solutions, implemented by assigning a value to β from the interval $[0, 1)$, selecting $\beta = 0.4$ in our experiment.

3.2 Variable Fixing Strategy

In order to describe the variable fixing and freeing strategies more precisely, the following is useful. Let F denote the index set for the fixed variables and U the index set for the free (unfixed) variables. Note that F and U partition the index set $N = \{1, \ldots, n\}$, i.e., $F \cup U = N$, $F \cap U = \emptyset$. Let na be the number of variables to be added when new variables are fixed and nd the number of variables to be dropped when new variables are freed. In addition, let x_i^F, for $i \in F$, denote the current values assigned to the fixed variables, and let x^s denote the starting solution at each run of TS. Then, at each iteration our algorithm begins by setting: $x_i^s = x_i^F$ for $i \in F$ and $x_i^s = Rand[0,1]$ for $i \in U$ and the tabu search procedure is launched to optimize this constructed initial solution. The two variable fixing strategies are described as follows:

Variable Fixing Strategy 1 (FIX1):
Order the elements of $i \in U$ such that $score(i_1) \leq \ldots \leq score(i_{|U|})$
Let $F(+) = i_1, \ldots, i_{na}$
$F := F \cup F(+)$ $(|F| := |F| + na)$
$U := U - F(+)$ $(|U| := |U| - na)$
$x_i^F = x_i^0$ for $i \in F(+)$, (x_i^F is already determined for $i \in$ "$previousF$" $:= F - F(+)$ and x_i^0 represents the value that x_i should be assigned to according to Eq. (6), i.e., $x_i^0 = 0$ if $VC_i(P:0) < VC_i(P:1)$ and $x_i^0 = 1$ otherwise.)

Variable Freeing Strategy 1 (FREE1):
Order the elements of $i \in F$ such that $score(i_1) \geq \ldots \geq score(i_{|F|})$
Let $F(-) = i_1, \ldots, i_{nd}$
$F := F - F(-)(|F| := |F| - nd)$
$U := U \bigcup F(-)(|U| := |U| + nd)$

Variable Fixing Strategy 2 (FIX2):
Set $|F| := |F| + na$
Order the elements of $i \in N$ such that $score(i_1) \leq \ldots \leq score(i_n)$
(We only need to determine the first $|F|$ elements of this sorted order.)
Let $F = i_1, \ldots, i_{|F|}$
$U := N - F(|U| := |U| - na)$
$x_i^F = x_i^0$ for $i \in F$

Variable Freeing Strategy 2 (FREE2):
Set $|F| := |F| - nd$
Order the elements of $i \in N$ such that $score(i_1) \leq \ldots \leq score(i_n)$
(We only need to determine the first F elements of this sorted order.)
Let $F = i_1, \ldots, i_{|F|}$
$U := N - F(|U| := |U| + nd)$
$x_i^F = x_i^0$ for $i \in F$

The strategy FIX1 differs in two ways from FIX2. At each fixing phase, FIX2 fixes $|F|$ variables, while FIX1 only fixes na new variables since $|F|-na$ variables are already fixed. In other words, once a variable is fixed by the strategy FIX1, its value cannot be changed unless a freeing phase frees this variable. Instead of inheriting the previously fixed variable assignment as in FIX1, FIX2 selects all $|F|$ variables to be fixed at each fixing phase.

In the freeing phase, the strategy FREE1 only needs to score variables belonging to F and then to select those with the highest scores to be freed, while FREE2 redetermines the variables to be freed each time.

3.3 Four Derived Algorithms

Our four key variants of the TS/VFS algorithm consist of the combination of the two variable fixing strategies and the two variable scoring strategies. Specifically, using $\beta = 1.0$ as our scoring strategy, we employ the variable fixing strategies FIX1 and FIX2 to get the first two algorithms, respectively. Likewise, the third and fourth algorithms are derived by combining the scoring strategy $\beta = 0.4$ with FIX1 and FIX2, respectively.

4 Experimental Results

4.1 Instances and Experimental Protocol

To evaluate the variable scoring and fixing strategies, we test the four variants of the TS/VFS algorithm on a set of 21 large and difficult random instances with 3000 to 7000 variables from the literature [9]. These instances are known to be much more challenging than those from ORLIB. Our algorithm is programmed in C and compiled using GNU GCC on a PC running Windows XP with Pentium 2.66GHz CPU and 512MB RAM. Given the stochastic nature of the algorithm, problem instances are independently solved 20 times. The stop condition for a single run is respectively set to be 5, 10, 30, 30, 50 minutes on our computer for instances with 3000, 4000, 5000, 6000 and 7000 variables, respectively.

4.2 Computational Results

We present in Tables 1 and 2 the computational results with β equaling to 1.0 and 0.4, respectively. Each table reports the results of both FIX1 and FIX2 variable fixing strategies. Columns 2 gives the density (dens) and Column 3 gives the best known objective values (f^*) obtained by all previous methods applied to these problems, as reported in [4, 7]. The remaining columns give the results of one of the two versions (FIX1 and FIX2) according to four criteria: (1) the best solution gap, g_{best}, to the previous best known objective values (i.e., $g_{best} = f^* - f_{best}$ where f_{best} denotes the best objective value obtained by our algorithm), (2) the average solution gap, g_{avr}, to the previous best known objective values (i.e., $g_{avr} = f^* - f_{avr}$ where f_{avr} represents the average objective

Table 1. Results of TS/VFS algorithms with variable fixing strategies FIX1 and FIX2 ($\beta = 1.0$)

Instance	dens	f^*	FIX1					FIX2					Sd
			g_{best}	g_{avr}	suc	t_{avr}	t_{best}	g_{best}	g_{avr}	suc	t_{avr}	t_{best}	
p3000.1	0.5	3931583	0	413	15	172	40	0	3193	5	54	63	Y
p3000.2	0.8	5193073	0	0	20	62	2	0	397	12	26	5	Y
p3000.3	0.8	5111533	0	71	18	115	6	0	1144	2	43	4	Y
p3000.4	1.0	5761822	0	114	16	93	5	0	3119	7	61	7	Y
p3000.5	1.0	5675625	0	372	8	86	5	0	1770	2	147	16	Y
p4000.1	0.5	6181830	0	0	20	65	14	0	319	19	74	16	N
p4000.2	0.8	7801355	0	1020	11	295	64	0	2379	5	81	59	Y
p4000.3	0.8	7741685	0	181	18	201	17	0	1529	9	58	20	Y
p4000.4	1.0	8711822	0	114	18	171	56	0	1609	9	209	39	Y
p4000.5	1.0	8908979	0	1376	9	231	58	0	2949	2	231	134	Y
p5000.1	0.5	8559680	0	670	1	999	999	368	2429	0	1800	1800	Y
p5000.2	0.8	10836019	0	1155	6	740	47	582	2528	0	1800	1800	Y
p5000.3	0.8	10489137	0	865	3	1037	279	354	4599	0	1800	1800	Y
p5000.4	1.0	12252318	0	1172	3	1405	1020	608	4126	0	1800	1800	Y
p5000.5	1.0	12731803	0	268	13	1003	192	0	2941	3	588	279	Y
p6000.1	0.5	11384976	0	914	6	451	68	0	4694	4	550	209	Y
p6000.2	0.8	14333855	0	1246	1	739	739	88	3332	0	1800	1800	Y
p6000.3	1.0	16132915	0	2077	2	1346	1267	2184	8407	0	1800	1800	Y
p7000.1	0.5	14478676	0	2315	1	2470	2470	744	4155	0	3000	3000	Y
p7000.2	0.8	18249948	716	2340	0	3000	3000	2604	6164	0	3000	3000	Y
p7000.3	1.0	20446407	0	2151	7	981	478	0	8150	5	1836	149	Y
Average			34	897	9.3	746	516	359	3330	4.0	988	848	Y

value), (3) the success rate, suc, for reaching the best result f^* and (4) the CPU time, consisting of the average time and the best time, t_{avr} and t_{best} (in seconds), for reaching the best result f^*. The last column Sd indicates the superiority of FIX1 over FIX2 when a 95% confidence t-test is performed in terms of the objective values. Furthermore, the last row "Average" indicates the summary of the algorithm's average performance.

Table 1 shows the computational results of variable fixing strategies FIX1 and FIX2 where $\beta = 1.0$. One observes that for all the considered criteria, FIX1 outperforms FIX2 for almost all the instances. Specifically, FIX1 is able to reach the previous best known objectives for all instances except one (p7000.2) while FIX2 fails for 8 cases. Moreover, FIX1 has an average success rate of 9.3 over 20 runs, more than two times larger than FS2's 4.0. FIX1 is also superior to FIX2 when it comes to the average gap to the best known objective values. In addition, FIX1 performs slightly better than FIX2 in terms of the CPU time to reach the best values. The T-test also demonstrates that FIX1 is significantly better than FIX2 except only one case (p4000.1).

Table 2 gives the computational results of variable fixing strategies FIX1 and FIX2 when β is set to be 0.4 instead of 1.0. From Table 2, we observe that FIX1 outperforms FIX2 in terms of all the considered criteria, including g_{best}, g_{avr}, suc, t_{avr} and t_{best}. One also notices that this is quite similar to the case of $\beta = 1.0$. Therefore, we can conclude that the variable fixing strategy FIX1 is generally superior to FIX2 when using the two variable scoring strategies considered in this paper. In other words, the two variable scoring strategies have a similar influence on the computational results. The ability of the tabu search method using FIX1 to obtain all of the best known solutions in the literature places this method on a par with the best methods like [4, 7], while its solution times are better than those obtained in [4].

Table 2. Results of TS/VFS algorithms with variable fixing strategies FIX1 and FIX2 ($\beta = 0.4$)

Instance	dens	f^*	FIX1					FIX2					Sd
			g_{best}	g_{avr}	suc	t_{avr}	t_{best}	g_{best}	g_{avr}	suc	t_{avr}	t_{best}	
p3000.1	0.5	3931583	0	308	16	98	4	0	3315	5	75	2	Y
p3000.2	0.8	5193073	0	0	20	59	9	0	488	13	50	3	Y
p3000.3	0.8	5111533	0	166	17	108	2	0	1355	4	28	5	Y
p3000.4	1.0	5761822	0	19	19	109	24	0	1684	10	74	2	Y
p3000.5	1.0	5675625	0	275	11	147	14	0	1796	3	154	40	Y
p4000.1	0.5	6181830	0	0	20	61	13	0	354	19	78	3	N
p4000.2	0.8	7801355	0	783	11	369	44	0	2722	3	382	106	Y
p4000.3	0.8	7741685	0	254	17	234	29	0	1474	8	75	29	Y
p4000.4	1.0	8711822	0	75	19	250	13	0	2537	7	158	12	Y
p4000.5	1.0	8908979	0	1769	8	361	275	0	3112	3	101	41	Y
p5000.1	0.5	8559680	0	791	2	721	228	325	2798	0	1800	1800	Y
p5000.2	0.8	10836019	0	860	4	540	37	0	2397	1	45	45	Y
p5000.3	0.8	10489137	0	1698	5	702	292	354	4939	0	1800	1800	Y
p5000.4	1.0	12252318	0	1123	2	103	76	444	3668	0	1800	1800	Y
p5000.5	1.0	12731803	0	455	12	747	261	0	3250	3	145	114	Y
p6000.1	0.5	11384976	0	1450	9	1014	432	0	5405	2	1178	768	Y
p6000.2	0.8	14333855	0	1079	3	911	515	0	4923	1	192	192	Y
p6000.3	1.0	16132915	0	2320	3	1000	642	0	6137	1	147	147	Y
p7000.1	0.5	14478676	0	1784	2	1519	785	1546	4556	0	3000	3000	Y
p7000.2	0.8	18249948	0	2743	1	2238	2238	1710	5986	0	3000	3000	Y
p7000.3	1.0	20446407	0	3971	3	1457	870	0	11604	1	1113	1113	Y
Average			0	1044	9.7	607	324	209	3548	4.0	733	668	Y

5 Discussion and Analysis

We now turn our attention to discussing and analyzing some key factors which may explain the performance difference of algorithms when using different variable fixing and scoring strategies. For this purpose, we examine the Variables Fixing Errors (number of wrongly fixed variables) relative to the putative optimal solution and show a fitness landscape analysis of high-quality solutions.

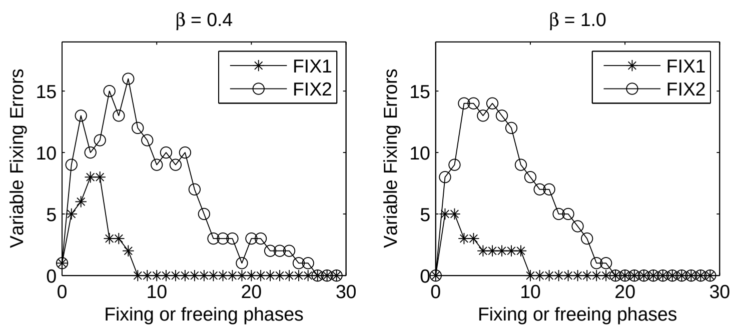

Fig. 1. Comparison of variable fixing errors between two fixing strategies

5.1 Variable Fixing Errors

As previously demonstrated, the variable fixing strategy FIX1 dominates FIX2 with both scoring strategies (with $\beta = 1.0$ and $\beta = 0.4$). In order to ascertain why this is the case, we conduct an experiment to compare the total number of wrongly fixed variables during the search using these two variable fixing strategies. For this, we carry out our experiment on instance p5000.5 and repeat the experiment 20 times. For each run, we count, after each fixing or freeing phase, the number of mismatched variables of the current (possibly partial) solution with respect to the best known solution[3]. Figure 1, where each point represents the accumulated Variable Fixing Errors over 20 runs, shows how the variable fixing strategies affect the Variable Fixing Errors at each fixing or freeing phase under two variable scoring strategies: the left one is for $\beta = 0.4$ and the right is for $\beta = 1.0$. From Figure 1, one observes that the number of variable fixing errors induced by FIX1 and FIX2 (with both scoring strategies) increases rapidly at the beginning of the search and then decreases gradually when the search progresses. However, the number of the Variable Fixing Errors of FIX1 is much smaller than that of FIX2 throughout the search process. This observation together with the results in Tables 1 and 2 demonstrate that the variable fixing strategy plays a vital role in our TS/VFS algorithm for both $\beta = 1.0$ and $\beta = 0.4$.

5.2 Fitness Distance Correlation Analysis

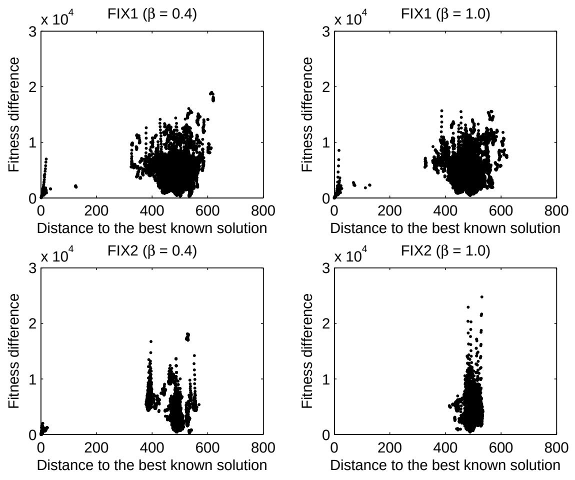

Fig. 2. Fitness distance correlation: instance p5000.1

In this section, we show a search landscape analysis using the fitness distance correlation [5], which estimates how closely the fitness and distance are related to the nearest optimum in the search space. For this purpose, we collect

[3] The best known solutions are obtained by different algorithms, sharing exactly the same assignment. Thus, we assume that it is very likely to be the optimal solution.

a large number of high-quality solutions by performing 20 independent runs of our TS/VFS algorithm, each run being allowed 30 fixing and freeing phases, where each phase has 20 elite solutions recorded in the population P. Thus, $20 * 30 * 20 = 12,000$ solutions are collected and plotted. Figures 2 and 3 show the hamming distance between these solutions to the best known solution against the fitness difference $\Delta f = f^*$ - $f(x^k)$ of these high-quality solutions for instances p5000.1 and p5000.5, respectively.

Figure 2 discloses that the majority of the high quality solutions produced by variable fixing strategy FIX1 (two upper sub-figures) has a much wider distance range than the solutions produced by strategy FIX2 (two bottom sub-figures), which indicates that the search space of FIX1 is more dispersed than that of FIX2. Moreover, the high-quality solutions of FIX1 are much closer to the x-axis than FIX2, implying that FIX1 can obtain better objective values than FIX2. In sum, this indicates the higher performance of the FIX1 strategy.

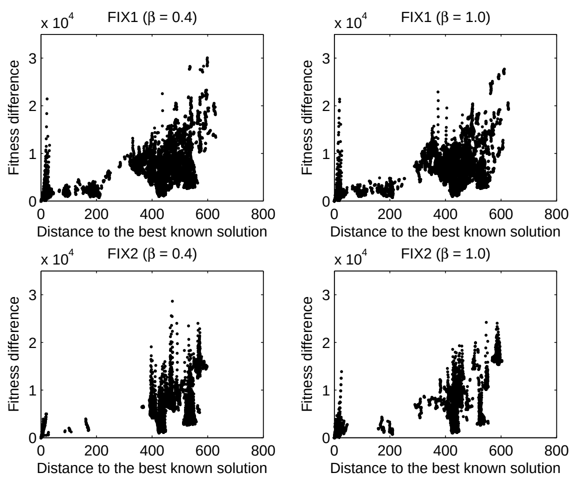

Fig. 3. Fitness distance correlation: instance p5000.5

Figure 3 presents a trend quite similar to that of Figure 2 in terms of the solutions' distance range and the percentage of high quality solutions when comparing the two variable fixing strategies FIX1 (two upper sub-figures) and FIX2 (two bottom sub-figures). However, a clear difference from Figure 2 is that high quality solutions are distributed in a wider range. In particular, the distribution of solutions is more continuous and does not produce the "isolated cluster effect" shown in Figure 2. This indicates that instance p5000.5 is much easier than p5000.1 to solve as shown in Tables 1 and 2. Indeed, for instance p5000.5, the search space seems smoother, enabling the search to traverse easily from solutions that are far from optimal to the best known solution.

6 Conclusions

To build better algorithms that make use of variable fixing and scoring, it is important to understand and explain the performance variations produced by different variable scoring and fixing strategies. We undertook to analyze the intrinsic characteristics of two variable fixing strategies and two variable scoring strategies for UBQP. To this end, we compared the Variable Fixing Errors produced in the course of obtaining a (near) optimal solution and identified the correlations between fitness distances of high quality solutions to characterize the search behavior of the variable fixing and scoring strategies. Our experimentation discloses that our TS method indeed performs differently according to the variable fixing strategy employed, but is much less sensitive to the variable scoring strategy. The finding that the fixing strategy FIX1 obtains the best solutions in the literature to the challenging test problems examined underscores the relevance of variable fixing strategies and the value of analyzing their impacts.

Acknowledgement

We are grateful for the referees for their comments and questions which helped us to improve the paper. The work is partially supported by "Pays de la Loire" Region (France) through RaDaPop and LigeRO projects (2009-2013).

References

1. Glover F. (1977) Heuristics for Integer Programming Using Surrogate Constraints. Decision Sciences 8(1):156–166
2. Glover F., Kochenberger G.A., Alidaee B. (1998) Adaptive memory tabu search for binary quadratic programs. Management Science 44:336–345
3. Glover F., Laguna M. (1997) Tabu Search. Kluwer Academic Publishers, Boston
4. Glover F., Lü Z., Hao J.K. (2010) Diversification-driven tabu search for unconstrained binary quadratic problems. 4OR 8(3): 239–253
5. Jones T., Forrest S. (1995) Fitness Distance Correlation as a Measure of Problem Difficulty for Genetic Algorithms. *Proceedings of the 6th International Conference on Genetic Algorithms*, Morgan Kaufmann, 184–192
6. Kochenberger G.A., Glover F., Alidaee B., Rego C. (2004) A unified modeling and solution framework for combinatorial optimization problems. OR Spectrum 26:237-250
7. Lü Z., Glover F., Hao J.K. (2010) A Hybrid Metaheuristic Approach to Solving the UBQP Problem. European Journal of Operational Research 207(3): 1254–1262
8. Merz P., Katayama K. (2004) Memetic algorithms for the unconstrained binary quadratic programming problem. BioSystems 78:99-118
9. Palubeckis G. (2004) Multistart tabu search strategies for the unconstrained binary quadratic optimization problem. Annals of Operations Research 131:259-282
10. Wilbaut C., Salhi S., Hanafi S. (2009) An iterative variable-based fixation heuristic for 0-1 multidimensional knapsack problem. European Journal of Operation Research 199:339–348
11. Zhang W. (2004) Configuration landscape analysis and backbone guided local search: Satisfiability and maximum satisfiability. Artificial Intelligence 158:1–26