INTERNATIONAL TRANSACTIONS IN OPERATIONAL RESEARCH

INTERNATIONAL TRANSACTIONS IN OPERATIONAL RESEARCH

Intl. Trans. in Op. Res. 21 (2014) 127–152
DOI: 10.1111/itor.12043

An analysis of parameter adaptation in reactive tabu search

Franco Mascia[a], Paola Pellegrini[b], Mauro Birattari[a] and Thomas Stützle[b]

[a]*IRIDIA, CoDE, Université Libre de Bruxelles, Brussels, Belgium*
[b]*IFSTTAR—ESTAS, Université Lille Nord de France, Lille, France*
E-mail: fmascia@ulb.ac.be [Mascia]; paola.pellegrini@ifsttar.fr [Pellegrini]; mbiro@ulb.ac.be [Birattari]; stuetzle@ulb.ac.be [Stützle]

Received 11 January 2013; received in revised form 20 June 2013; accepted 20 June 2013

Abstract

On-line parameter adaptation schemes are widely used in metaheuristics. They are sometimes preferred to off-line tuning techniques for two main reasons. First, they promise to achieve good performance even on new instance families that have not been considered during the design or the tuning phase of the algorithm. Second, it is assumed that an on-line scheme could adapt the algorithm's behaviour to local characteristics of the search space. This paper challenges the second hypothesis by analysing the contribution of the parameter adaptation to the performance of a state-of-the-art reactive tabu search (RTS) algorithm for the maximum clique problem. Our experimental analysis shows that this on-line parameter adaptation scheme converges to good instance-specific settings for the parameters, and that there is no evidence that it adapts to the local characteristics of the search space. The insights gained from the analysis are confirmed by further experiments with an RTS algorithm for the quadratic assignment problem. Together, the results of the two algorithms shed some new light on the reasons behind the effectiveness of RTS.

Keywords: On-line parameter adaptation; reactive tabu search; maximum clique problem; quadratic assignment problem

1. Introduction

Optimisation problems arise in many areas of science and engineering. The recent years have seen an explosion of stochastic local search (SLS) methods such as tabu search (Glover, 1986; Hansen and Jaumard, 1990), memetic algorithms (Moscato, 1999), iterated local search (Lourenço et al., 2010), ant colony optimisation (Dorigo et al., 1991; Dorigo and Stützle, 2004), and many others for tackling NP-hard optimisation problems. These methods are often characterised by a large number of parameters that allow selecting and fine tuning of algorithmic components. Unfortunately, the performance of these SLS methods depends strongly on the parameter settings, and proper parameter settings change strongly with different problems or instances of a given problem. As

a possible solution to this problem, on-line parameter adaptation schemes have been proposed (Eiben et al., 2007; Stützle et al., 2012). A prominent representative of such schemes is reactive tabu search (RTS; Battiti and Tecchiolli, 1994). RTS is an SLS method that uses the search history for adapting the length of the tabu list—a parameter that is known to be crucial for tabu search performance.

RTS is a part of the wider framework of reactive local search (Battiti et al., 2008), which advocates, in general, the use of memory for the on-line adaptation of parameters that control the tradeoff between intensification and diversification in SLS algorithms. Our study focusses on RTS as it is a well-known algorithm. In fact, after the seminal paper of Battiti and Tecchiolli (1994), RTS has been successfully applied to several optimisation problems (Bastos and Ribeiro, 2001; Battiti and Bertossi, 1999; Chiang and Russell, 1997; Nanry and Wesley Barnes, 2000; Osman and Wassan, 2002; Ryan et al., 1998; Toune et al., 1998).

RTS and, more in general, on-line parameter adaptation schemes are claimed to offer the following two advantages over off-line tuning techniques. First, there is no need for an extensive tuning phase before these techniques can be deployed; therefore, in principle, these techniques could be effective when applied to entirely new instance classes having properties that have not been observed during the design or tuning phase. Second, while an SLS algorithm navigates the search space by moving from a candidate solution to a neighbouring one, a parameter adaptation scheme can adjust the algorithm's parameters to local characteristics of the search space; this is a property that intuitively should give the adaptation scheme an advantage over a parameter setting that is kept fixed during the search. Nevertheless, deep insight into the algorithm is necessary to design an effective adaptation scheme; it is in fact essential to know which is the most crucial, or which are the few most important parameters to adapt, and how to adapt them.

In this work, we analyse the dynamics of RTS-MCP (Battiti and Mascia, 2010), a recent state-of-the art RTS algorithm, for a prominent combinatorial optimisation problem—the maximum clique problem (MCP). We confirm the insight obtained in the study by analysing RTS for the quadratic assignment problem (QAP), which is the first RTS algorithm proposed in the literature (Battiti and Tecchiolli, 1994), and is a state-of-the-art algorithm for specific instance classes. We will refer to this algorithm as RTS-QAP.

In this study, we aim to seek evidence in favour or against the following two hypotheses: the first hypothesis is that RTS is able to adapt the tabu list length parameter to a good instance-wise setting; the second is that RTS also adapts to local characteristics of the search space. To study the first hypothesis, in Section 2.3, we present an analysis on the best instance-wise parameter setting for the benchmark instances used in this study. Then, in Section 2.4, we analyse the dynamics of the parameter adaptation scheme of RTS-MCP. To study the second hypothesis, in Section 2.5, we compare RTS-MCP with a variant having a fixed setting of the tabu list length. To further strengthen the results, we also compare, in Section 2.6, RTS-MCP with a variant that sets, at each step, the value of the tabu list length to a random value. Based on the analysis of the previous sections, we present, in Section 2.7, a robust tabu search for the MCP, as a simple alternative tabu search algorithm that reaches high performance without on-line parameter adaptation. Note, however, that the goal of the paper is not to improve the state of the art, but to analyse the reasons underlying the high performance of RTS and understand its working principles. The results we obtained for

RTS-MCP are confirmed in Section 3 by an analogous analysis on RTS-QAP. In Section 4, we draw the conclusions.

2. The maximum clique problem

A clique in graph $G = (V, E)$ is a subset $S \in V$, in which all nodes are pairwise adjacent, that is, $\forall u, v \in S \; \exists (u, v) \in E$. The goal of MCP is to find a clique of maximum cardinality.

MCP is a well-studied NP-hard combinatorial optimisation problem (Balas and Yu, 1986; Bomze et al., 1999) with important applications in data mining (Boginski et al., 2006), computer vision (Pla and Marchant, 1997), social network analysis (Palla et al., 2007), computational biochemistry (Butenko and Wilhelm, 2005), bio-informatics (Ji et al., 2004; Mascia et al., 2010), genomics (Pevzner and Sze, 2000) and biological networks (Adamcsek et al., 2006; Voy et al., 2006). The problem received a lot of attention in the literature, and it was one of the problems of the second DIMACS implementation challenge (Johnson et al. 1996; http://dimacs.rutgers.edu/Challenges). Also, in the past few years before writing this paper, several new SLS algorithms for the MCP have been proposed (Battiti and Mascia, 2010; Grosso et al., 2007; Pullan, 2006; Pullan and Hoos, 2006; Pullan et al., 2011).

2.1. RTS-MCP

RTS-MCP is a state-of-the-art RTS algorithm that was originally proposed by Battiti and Protasi (2001) and further improved by Battiti and Mascia (2010). The algorithm goes through a series of greedy constructions, and diversifies by means of a tabu search and frequent restarts.

In RTS-MCP, the neighbourhood of a candidate solution amounts to all cliques that can be reached by adding or dropping a node from the current solution. A data structure called PossibleAdd contains the nodes that are adjacent to all nodes of the current solution. RTS-MCP starts by selecting a random node in the graph and initialising the set PossibleAdd accordingly. Successively, the algorithm constructs a candidate solution by adding nodes from PossibleAdd. During this construction, nodes are selected among those having the highest degree in the subgraph induced by PossibleAdd. Ties are broken randomly. In this way, RTS-MCP tries to add the nodes that in the successive steps lead to the least reduction of the size of PossibleAdd. When PossibleAdd is empty, RTS-MCP has reached a local optimum; at this point, the algorithm removes a node from the current solution by selecting randomly among those that when dropped lead to the maximum increase of the size of PossibleAdd. RTS-MCP alternates between these constructions and node removals, and ensures diversification by means of a tabu search. Each time a node is added or dropped from the current solution, the move cannot be undone for the successive T steps, where T is a parameter of the algorithm. Moreover, if the best solution found so far is not improved for a number of steps larger than R times the size of the largest clique found so far, the algorithm is restarted. In RTS-MCP, the restart parameter R is fixed to 100.

The length T of the tabu list is a crucial parameter for the performance of tabu search since it controls the amount of intensification and diversification of the algorithm. The tabu list length is initialised to 1, and successively adapted on-line by leveraging the history of the search. All

cliques encountered during the search are stored in a data structure; every time a cycle is detected, that is, the same solution is revisited within $2 \cdot (\mathsf{numNodes} - 1)$ steps, the tabu list length T is set to $\max\{T + 1, T \cdot 1.1\}$. If, since the last update of the parameter T, no repeated solutions are encountered for a number of steps greater than $20 \cdot \mathsf{SizeBestCliqueSoFar}$, the tabu list length is set to $\min\{T - 1, T \cdot 0.9\}$. Since too many repetitions could lead to an explosion of the values taken by the parameter, the algorithm sets a threshold $\mathrm{MAXT} = \mathsf{SizeBestCliqueSoFar} + 0.5$. Therefore, the tabu list length can only take values in the interval [1, MAXT]. See the paper by Battiti and Mascia (2010) for more details.

2.2. The benchmark set

The commonly used benchmark set for the MCP in the literature was proposed about 20 years ago in the second DIMACS Implementation Challenge (1992–1993; http://dimacs.rutgers.edu/Challenges). Since the benchmark set has been around for so many years, SLS algorithms have become very effective in solving these instances. The benchmark set is composed of

- the C and DSJC families with instances ranging from 125 to 4000 nodes;
- the Brockington–Culberson (brock) family with instances ranging from 200 to 800 nodes; these random instances have been created by hiding the maximum clique among nodes that have a relatively low degree (Brockington and Culberson, 1996);
- the Mannino (MANN_a) family with instances ranging from 378 to 3321 nodes; these instances are generated starting from set covering problems on Steiner triple graphs;
- the Keller (keller) family, with instances ranging from 171 to 3361 nodes; these instances are based on Keller's conjecture on tilings using hypercubes;
- other random instances ranging from 200 to 1500 nodes; these instances belong to the gen, hamming and p_hat families, and are usually solved to optimality in few milliseconds.

From the subset of instances that are commonly used to compare algorithms for the MCP, we remove the keller4 and the hamming8-4 instances as they are typically solved in the first construction. In the Appendix, we present a new best solution of size 1100 for the instance MANN_a81. To the best of our knowledge, this is the first time a solution of size 1100 is presented in the literature.

Table 1 summarises the properties of the instances in the benchmark set. The smallest among these instances can be solved in few thousand steps of RTS-MCP, which, on a desktop computer with a CPU running at 2 GHz, translates to less than 1 millisecond of CPU-time. Comparing algorithms based on their CPU-time becomes difficult because of the limited resolution in the functions used for measuring CPU-time. Despite the issues with these small instances, this benchmark set is still the standard one used to compare algorithms on MCP, and there still remain some hard instances such as C2000.9, MANN_a45, MANN_a81 and the brock800, where finding the best-known solutions takes more than tens of millions of steps. To present the results uniformly on such a heterogeneous benchmark set, we compare RTS-MCP and its variants on the number of steps to reach the best-known solution. For the instances for which no optimum is known, we considered the best-known

solution in the literature. For the detailed results on the CPU-times measured on the reference machine,[1] we refer to the supplementary pages of Mascia et al. (2011).

2.3. Parametric study of the restart parameter and tabu list length

We measure the impact that the restart parameter and tabu list length have on the number of steps to find the best-known solutions. Two exemplary results are shown in Fig. 1. Figure 1 shows the median number of steps to find the best-known solution for $\mathsf{Fixed}_{\mathsf{TL}}$-MCP, which corresponds to RTS-MCP where the reactive search mechanism is disabled and one fixed tabu list length is used throughout the whole run of the algorithm. For a detailed presentation of the results on all instances, we refer to the supplementary pages of Mascia et al. (2011). Each point in the plots in Fig. 1 corresponds to the median number of steps to reach the best-known solution on instances keller6 (Fig. 1(a)) and MANN_a27 (Fig. 1(b)) over 10 runs for a specific combination of a fixed tabu list length T and a fixed restart parameter R. We set the maximum number of steps to 10^7; therefore, the combinations of values of T and R for which the algorithm did not find the best-known solution within the maximum number of steps are all plotted at 10^7 steps.

What emerges from the plot in Fig. 1(a) is that, except for very small values of R, there is a large interval of good values for the tabu list length from approximately 20 to 30. Among the two parameters driving diversification, the tabu list length is the most critical one for the keller6 instance. The restarts are useful because they add overall robustness to the algorithm. The relative importance of the tabu list length over the restart parameter is confirmed also on other hard instances in the benchmark set (C1000.9, C2000.5, C2000.9, C4000.5, brock200_4, brock400_4, keller5, p_hat1500-1). There is only one notable exception to the aforementioned relative importance of the parameters, that is, the Mannino family of instances. These instances are characterised by large plateaus, and few repeated solutions are encountered during the search (Pullan et al., 2011); in this case, the restart parameter is more important (see Fig. 1(b)), while the tabu list length is not as important. Plots for instances MANN_a45 and MANN_a81 are shown in the supplementary pages of Mascia et al. (2011). For these two large instances, the plot represents the median number of steps to reach non-optimal solutions of size 344 and 1098, respectively.

2.4. The parameter adaptation

The question now is whether the reactive mechanism is able to spot the large interval of good settings for the tabu list length. To answer this, we first examine how the tabu list length evolves during the search. Then we check how the empirical distribution of the tabu list length matches the region of good values.

Figure 2 shows the values of the tabu list length set by RTS-MCP during two typical runs on instance keller6 (Fig. 2(a)) and instance brock200_2 (Fig. 2(b)). On keller6, there is a quick

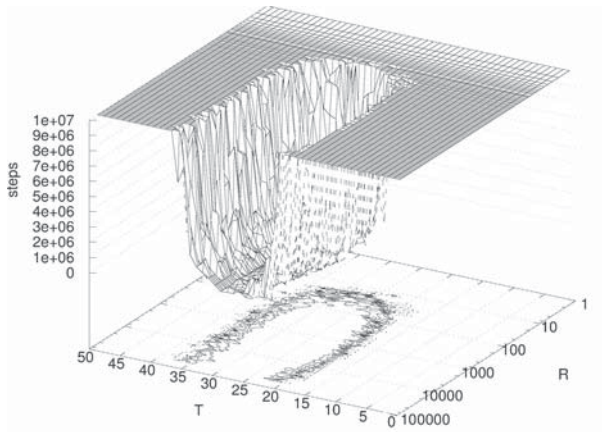
[1]CPU-time is measured on a single core of a cluster of Intel Xeon Quad-core processors, running at 2.33 GHz, and with 500 MB of RAM devoted to each process. The cluster runs under the Rocks Cluster 5.3 distribution, which is based on CentOS 5.3 Linux.

Table 1
Selected DIMACS instances with bound ($\omega(G)$), best-known or optimal solution size, number of nodes, number of edges, median (and interquartile range) of the node degree distribution, median (and interquartile range) of the node degree distribution of the optimal or best-known solution

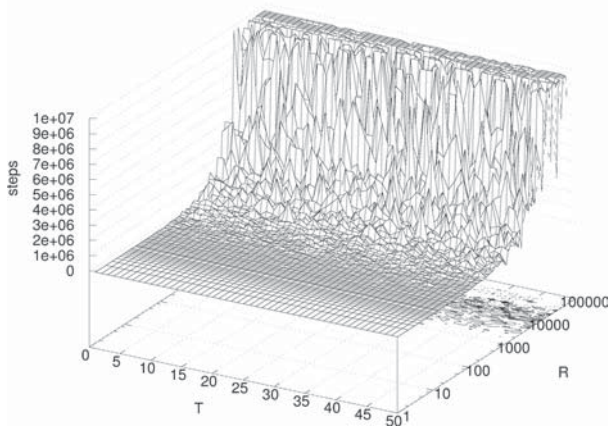
Instance	$\omega(G)$	Best	Number of nodes	Number of edges	Graph degree distribution		Best degree distribution	
C125.9	≥34	34	125	6963	112.0	(5.00)	114.5	(4.75)
C250.9	≥44	44	250	27,984	224.0	(6.00)	227.0	(5.00)
C500.9	≥57	57	500	112,332	449.0	(9.00)	455.0	(9.00)
C1000.9	≥68	68	1000	450,079	900.0	(13.00)	907.0	(11.25)
C2000.9	≥80	80	2000	1,799,532	1800.0	(18.00)	1803.0	(15.25)
DSJC1000_5	15	15	1000	499,652	500.0	(20.00)	503.0	(23.00)
DSJC500_5	13	13	500	125,248	250.0	(16.00)	259.0	(14.00)
C2000.5	≥16	16	2000	999,836	999.0	(30.00)	1006.0	(11.50)
C4000.5	≥18	18	4000	4,000,268	2001.0	(42.00)	2002.0	(41.00)
MANN_a27	126	126	378	70,551	374.0	(0.00)	374.0	(0.00)
MANN_a45	345	345	1035	533,115	1031.0	(0.00)	1031.0	(0.00)
MANN_a81	≥1100	1100	3321	5,506,380	3317.0	(0.00)	3317.0	(0.00)
brock200_2	12	12	200	9876	99.0	(10.00)	101.0	(11.00)
brock200_4	17	17	200	13,089	131.0	(8.00)	134.0	(6.00)
brock400_2	29	29	400	59,786	299.0	(10.00)	299.0	(9.00)
brock400_4	33	33	400	59,765	299.0	(11.00)	299.0	(9.00)
brock800_2	24	24	800	208,166	521.0	(18.00)	516.5	(20.25)
brock800_4	26	26	800	207,643	519.0	(18.25)	512.0	(20.25)
gen200_p0.9_44	44	44	200	17,910	180.0	(8.00)	179.5	(4.25)
gen200_p0.9_55	55	55	200	17,910	179.0	(7.25)	179.0	(5.50)
gen400_p0.9_55	55	55	400	71,820	360.0	(13.25)	359.0	(6.00)
gen400_p0.9_65	65	65	400	71,820	361.0	(14.00)	359.0	(9.00)
gen400_p0.9_75	65	65	400	71,820	359.0	(13.00)	359.0	(8.00)
hamming10-4	40	40	1024	434,176	848.0	(0.00)	848.0	(0.00)
keller5	27	27	776	225,990	578.0	(38.00)	578.0	(33.00)
keller6	≥59	59	3361	4,619,898	2724.0	(50.00)	2724.0	(50.00)
p_hat300-1	8	8	300	10,933	73.0	(39.00)	103.0	(20.00)
p_hat300-2	25	25	300	21,928	146.5	(73.00)	213.0	(18.00)
p_hat300-3	36	36	300	33,390	224.0	(38.00)	251.0	(15.25)
p_hat700-1	11	11	700	60,999	174.5	(87.00)	250.0	(22.50)
p_hat700-2	44	44	700	121,728	353.0	(177.50)	508.0	(31.50)
p_hat700-3	≥62	62	700	183,010	526.0	(89.00)	602.0	(14.00)
p_hat1500-1	12	12	1500	284,923	383.0	(197.00)	509.0	(82.00)
p_hat1500-2	≥65	65	1500	568,960	763.0	(387.00)	1100.0	(37.00)
p_hat1500-3	≥94	94	1500	847,244	1132.5	(192.00)	1297.5	(25.75)

Note: Known optimum solutions are those for which the bound $\omega(G)$ does not contain an inequality sign.

convergence to a specific level with further oscillations around it for the remaining steps. On brock200_2, the value of the tabu list length approaches quickly a first threshold MAXT = 11. Such a threshold effect is visible on hard instances, where the number of steps to converge to the best-known solution is large as in brock200_2 (Fig. 2(b)), brock400_2 and brock800_4. For other hard instances, such as C2000.9, where the best-known solution is large, the threshold



(a) DIMACS instance keller6.

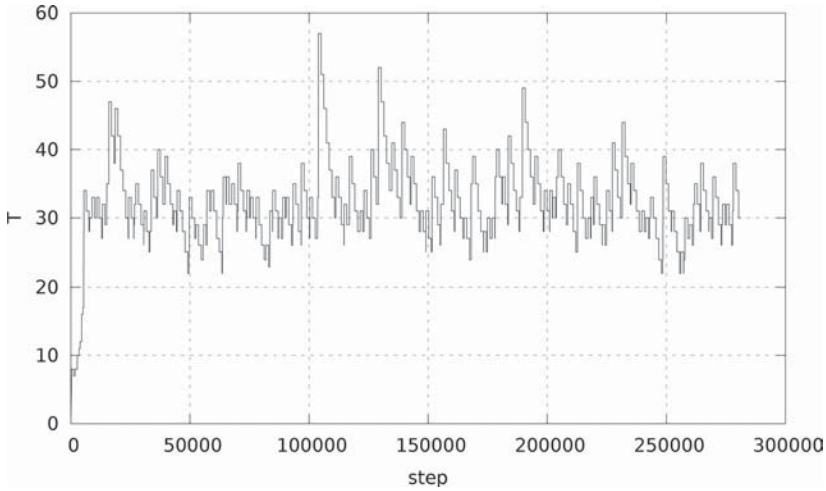


(b) DIMACS instance MANNa

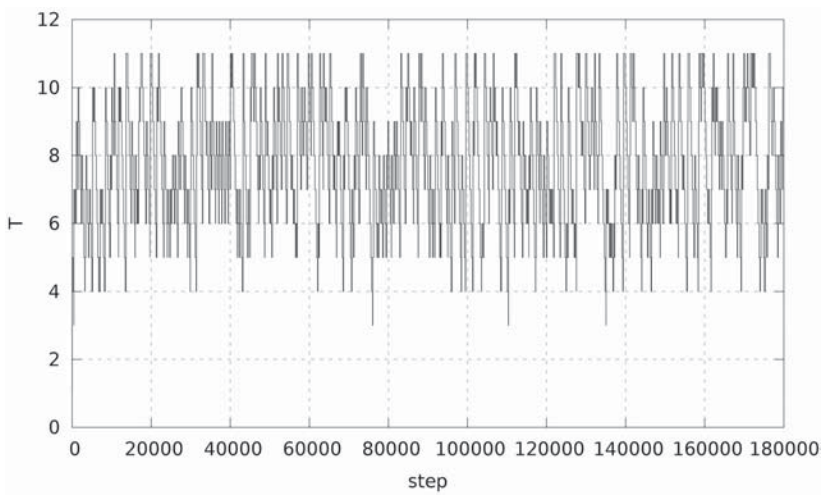
Fig. 1. Median number of steps to find the best-known solution across 10 runs of $\mathsf{Fixed}_{\mathsf{TL}}$-MCP for two instances of the DIMACS benchmark set.

MAXT is never reached. On most instances, however, there is a quick convergence to an instance-specific value with minor oscillations around it, following a behaviour analogous to that shown in Fig. 2(a).

To check if this instance-specific range of values is a good range of values for the instance at hand, in Fig. 3, we plotted the empirical distribution of the tabu list length values used by RTS-MCP against the median number of steps to reach a best-known solution with $\mathsf{Fixed}_{\mathsf{TL}}$-MCP for a range of tabu list length settings. The median number of steps to the best-known solution of $\mathsf{Fixed}_{\mathsf{TL}}$-MCP correspond to a slice in the bivariate plots in Fig. 1 for restart parameter R equal to 100, which is the value fixed in RTS-MCP. The median is computed over 100 runs. If the first hypothesis is true, we expect intuitively that the range of values taken by the tabu list length in RTS-MCP is close to the range of values around which $\mathsf{Fixed}_{\mathsf{TL}}$-MCP has the fastest



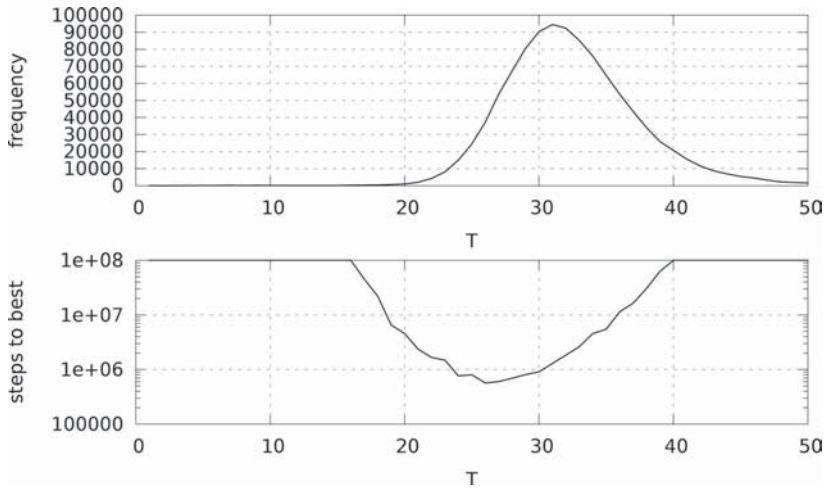
(a) DIMACS instance keller6.



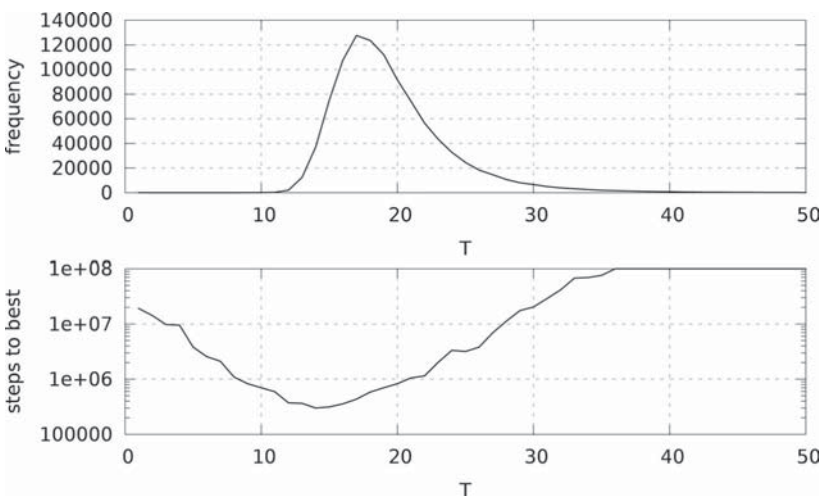
(b) DIMACS instance brock200_2.

Fig. 2. Evolution of the tabu list length during a run of RTS-MCP.

convergence. Figure 3(a) shows the adaptation of the tabu list length and the number of steps to the best-known solution for fixed tabu lengths on instance keller6; the mode of the distribution of values for T of RTS-MCP is 31, while the instance-optimal setting is 24. Figure 3(b) shows the same comparison of instance C1000.9. In this case, the mode of the empirical T distribution is 17 and the instance-optimal setting is 10. The plots for the other instances in the benchmark set are available in the supplementary pages of Mascia et al. (2011). Table 2 summarises, for each instance, the mean of the empirical T distributions (T_{mean}) of RTS-MCP and the optimal settings (T_{best}) of Fixed_{TL}-MCP. The values in T_{best} are the tabu list lengths for which we could not reject the null hypothesis of having no difference in the number of steps for finding the best-known solution when



(a) DIMACS instance keller6.



(b) DIMACS instance C1000.9.

Fig. 3. Comparison of the empirical distribution of the tabu list length and the number of steps to reach best-known solutions for fixed values of T. See the text for more details.

compared to the best setting. The statistical significance is assessed with a Wilcoxon rank-sum test with significance level $\alpha = 0.05$. Based on the values in Table 2, there are six instances where the average value taken by the tabu list length in RTS-MCP is in the range of optimal fixed settings, 11 cases where the average value is slightly larger and 17 cases where the average value is slightly smaller. In the latter case, with the exception of MANN_a45, MANN_a81 and large instances of the Brockington–Culberson family, the instances are rather easy to solve, and the on-line parameter adaptation scheme has not enough time to adapt the tabu list length before the best-known solution is found.

Table 2
Average value of the tabu list length set by the reactive mechanism of RTS-MCP (T_{mean}), compared with the instance-optimal settings (T_{best}) for Fixed$_{TL}$-MCP

Instance	T_{mean}	T_{best}
C125.9	6.9	[3, 17] \{4, 5, 6}
C250.9	15.4	[9, 20] \{10}
C500.9	19.6	[9, 15]
C1000.9	19.5	[12, 17]
C2000.9	*19.4*	*[18]*
DSJC1000.5	6.7	[3, 6]
DSJC500.5	6.7	[3, 5]
C2000.5	6.6	[3, 6]
C4000.5	*6.7*	*[3]*
MANN_a27	1.6	[2, 50] \{25, 46}
MANN_a45	*1.2*	*[26]*
MANN_a81	*1.2*	*[15]*
brock200_2	7.9	[10, 13]
brock200_4	10.0	[11, 15]
brock400_2	*11.3*	*[16]*
brock400_4	11.3	[13, 22]
brock800_2	*8.8*	*[15]*
brock800_4	*8.7*	*[17]*
gen200_p0.9_44	15.9	[16, 23]
gen200_p0.9_55	12.7	[18, 35] \{20}
gen400_p0.9_55	19.1	[16, 21] \{17}
gen400_p0.9_65	13.9	[18, 31]
gen400_p0.9_75	13.6	[26, 44] \{29, 31, 41, 42, 43}
hamming10-4	11.4	[7, 17]
keller5	19.2	[14, 16]
keller6	32.4	[24, 29] \{25}
p_hat300-1	5.2	[2, 4]
p_hat300-2	1.2	[1, 15] \{14}
p_hat300-3	15.5	[12, 16]
p_hat700-1	6.3	[3, 7]
p_hat700-2	3.0	[5, 13] \{6, 8}
p_hat700-3	7.5	[3, 16]
p_hat1500-1	6.1	[3, 6]
p_hat1500-2	16.6	[7, 30]
p_hat1500-3	13.3	[17, 28]

Note: Instances highlighted in italics are those that the two algorithms are not able to solve with 100% success rate.

2.5. *Fixed parameter settings*

As shown in the previous section, the reactive mechanism is able to spot a good value for the tabu list length in a short number of steps. In this section we analyse the second hypothesis, that is, whether this on-line parameter adaptation scheme also adapts the tabu list length to local characteristics of the search space. If this is the case, adapting the parameter might produce a measurable advantage over keeping the best fixed value throughout the search.

In Fixed$_{\mathsf{TL}}$-MCP, we fixed the tabu list length to the best value for the instance being optimised. For some hard instances such as C2000.9, MANN_a45, MANN_a81, brock400_2, brock800_2 and brock800_4, no fixed tabu list length setting was able to solve the instances with a 100% success rate. For these instances, we selected the tabu list length that resulted in the highest success rate. For MANN_a81, no fixed setting could find the best-known solution in any of the 100 runs, therefore we fixed the tabu list length to 1.

The comparison between RTS-MCP and Fixed$_{\mathsf{TL}}$-MCP is performed on the number of steps to reach the best-known solution and not on the computation time. Therefore, if there is a bias, then it is in favour of RTS-MCP, as in this way we do not take into consideration the extra CPU-time required for the maintenance of the search history in RTS-MCP, which can be costly during restart operations when the search history is re-initialised (Battiti and Mascia, 2010). Moreover, as detailed in Section 2.2, many instances are nowadays solved in few milliseconds, and using the actual algorithm steps allows spotting differences between two versions of the same algorithm more easily.

For every instance, we run 1000 experiments for a maximum number of steps that amounts to 10^8. We assess the statistical significance of the difference between the algorithms by means of a Wilcoxon rank-sum test with significance level $\alpha = 0.05$. The statistical tests have a high power since the empirical distributions compared are 1000 observations.

Table 3 presents the results of the comparison between RTS-MCP and Fixed$_{\mathsf{TL}}$-MCP. From this table, it is clear that on most instances, a fixed tabu list length requires less steps to find the best-known solution. On 25 of these instances, highlighted in boldface, a Wilcoxon rank-sum test rejected the null hypothesis at a 0.05 significance level. On the instances that neither RTS-MCP nor Fixed$_{\mathsf{TL}}$-MCP are able to solve in all runs, Fixed$_{\mathsf{TL}}$-MCP has a higher success rate, except for the C2000.9, MANN_a45 and MANN_a81 instances. Overall, the results suggest that if there is an advantage in adapting to the local characteristics of the search space, the reactive mechanism of RTS-MCP is not able to exploit it.

2.6. *Random parameter settings*

To strengthen this point, we changed Fixed$_{\mathsf{TL}}$-MCP to set at every step the value of the tabu list length to a random number, which is drawn with the same empirical distribution we observed on RTS-MCP. We call this algorithm Random$_{\mathsf{ED}}$-MCP. Setting the correct values for the local characteristics of the search space might give an advantage over setting them randomly. However, as shown in Table 4, the performance of the two algorithms is comparable. The only exception is the instance keller6, where the reactive setting leads to an average number of steps to the best-known solution that is one order of magnitude less than the random setting. On the reference machine, this translates to 50.6 CPU-seconds for Random$_{\mathsf{ED}}$-MCP against the 4.9 CPU-seconds of RTS-MCP. Given that on the other instances the performance of RTS-MCP and Random$_{\mathsf{ED}}$-MCP is rather similar, we investigated several hypotheses that could explain this large difference, but for the time being we could not find any convincing explanation.

If we set the tabu list length uniformly at random in the same interval [1, MAXT], from which the tabu list length is allowed to take values in RTS-MCP (Battiti and Mascia, 2010), we obtain

Table 3
Comparison of the median steps of RTS-MCP and Fixed_{TL}-MCP

Instance	RTS-MCP	Fixed_{TL}-MCP
C125.9	84.0	**64.0**
C250.9	1147.0	**904.0**
C500.9	81,144.0	**29,165.0**
C1000.9	708,530.0	**395,961.5**
C2000.9	***0.4%***	*0.2%*
DSJC1000.5	34,560.0	**23,535.0**
DSJC500.5	1400.5	**1166.0**
C2000.5	32,772.0	**19,001.5**
C4000.5	3,677,632.0	**2,569,921.0**
MANN_a27	75,262.0	75,262.0
MANN_a45	***0.2%***	*0.1%*
MANN_a81	***0.2%***	*0.0%*
brock200_2	56,583.0	**40,357.5**
brock200_4	178,136.0	**155,280.0**
brock400_2	*93.4%*	***99.1%***
brock400_4	1,699,627.0	**926,818.0**
brock800_2	*1.0%*	***2.1%***
brock800_4	*14.5%*	***21.6%***
gen200_p0.9_44	1429.0	**1059.0**
gen200_p0.9_55	584.0	**325.0**
gen400_p0.9_55	21,150.5	22,246.5
gen400_p0.9_65	1390.0	**917.0**
gen400_p0.9_75	1583.0	**674.0**
hamming10-4	491.0	507.0
keller5	3040.0	**1847.0**
keller6	672,768.0	**549,802.5**
p_hat300-1	139.0	**70.0**
p_hat300-2	27.0	27.0
p_hat300-3	620.0	**469.0**
p_hat700-1	1182.0	**836.0**
p_hat700-2	118.0	**95.0**
p_hat700-3	210.0	**186.0**
p_hat1500-1	139,617.0	**110,934.5**
p_hat1500-2	363.0	**277.0**
p_hat1500-3	1229.0	**649.0**

Note: Statistically significant improvements are highlighted in boldface. Instances highlighted in italics are those which one of the two algorithms was not able to solve with 100% success rate.

results that are fairly competitive with RTS-MCP. This is even more evident from the comparison of the CPU-seconds. This surprising result can be explained by the small size of the interval in which the reactive mechanism operates in this specific algorithm. The detailed results are reported in the supplementary pages of Mascia et al. (2011).

Table 4
Comparison of the median steps of RTS-MCP and $\text{Random}_{\text{ED}}$-MCP

Instance	RTS-MCP	$\text{Random}_{\text{ED}}$-MCP
C125.9	84.0	78.0
C250.9	1147.0	**906.0**
C500.9	**81,144.0**	100,647.0
C1000.9	708,530.0	**633,335.0**
C2000.9	***0.4%***	*0.0%*
DSJC1000.5	34,560.0	34,913.0
DSJC500.5	1400.5	1393.5
C2000.5	32,772.0	29,712.5
C4000.5	3,677,632.0	3,699,206.0
MANN_a27	75,262.0	75,262.0
MANN_a45	***0.2%***	*0.1%*
MANN_a81	***0.2%***	*0.0%*
brock200_2	56,583.0	53,017.0
brock200_4	178,136.0	180,913.0
brock400_2	***93.4%***	*92.7%*
brock400_4	1,699,627.0	1,625,401.0
brock800_2	*1.0%*	***1.3%***
brock800_4	***14.5%***	*14.0%*
gen200_p0.9_44	1429.0	**1163.0**
gen200_p0.9_55	584.0	536.0
gen400_p0.9_55	21,150.5	**18,252.0**
gen400_p0.9_65	1390.0	**1176.0**
gen400_p0.9_75	1583.0	**1274.0**
hamming10-4	491.0	566.0
keller5	**3040.0**	4611.5
keller6	**672,768.0**	7,052,179.0
p_hat300-1	139.0	**124.0**
p_hat300-2	27.0	27.0
p_hat300-3	620.0	**539.0**
p_hat700-1	1182.0	1269.0
p_hat700-2	118.0	114.0
p_hat700-3	210.0	**188.0**
p_hat1500-1	139,617.0	**115,778.5**
p_hat1500-2	363.0	**279.0**
p_hat1500-3	1229.0	**808.0**

Note: Statistically significant improvements are highlighted in boldface. Instances highlighted in italic are those which one of the two algorithms was not able to solve with 100% success rate.

2.7. *A robust tabu search*

Based on the insights gained from the analysis, this section shows how it is possible to replace the adaptation scheme and retain the performance without knowing *a priori* the best parameter setting for an instance at hand. The aim here is not to improve the state of the art but to offer a simple alternative to algorithms with on-line parameter adaptation.

The instances in the benchmark set belong to families having different properties, and the best parameter settings vary from family to family. Therefore, we try to model the relation between the instance-wise best fixed settings and the instance properties. We define a model for the tabu list length that depends on three easily measurable properties of the instance, that is, SizeBestCliqueSoFar, which serves as an estimation of the size of the maximum clique, the number of nodes and the number of edges in the graph:

$$\mathrm{pT} = c + \alpha\ \mathsf{SizeBestCliqueSoFar} + \beta\ \mathsf{numNodes} + \gamma\ \mathsf{numEdges}.$$

The data used for the regression consist of all instances except for brock400_2, brock800_2, brock800_4, MANN_a45, MANN_a81, C2000.9, which neither $\mathsf{Fixed}_{\mathsf{TL}}$-MCP nor RTS-MCP are able to solve to optimality with 100% success rate within 10^8 steps.

The data used to fit the model are tuples with the best tabu list length for an instance and the properties we measured on that instance. As shown in Table 2, the optimal setting for the tabu list length is not unique, therefore, for each instance, we sorted the tabu list lengths by the median number of steps to converge to the best-known solution, and we consider the first 10% of them for the data. We implemented a leave-one-out cross-validation, which means that for each instance the model has been fit on the data of all other instances. To fit the model, we resorted to bootstrapping: we randomly selected with replacement 1000 bootstrap samples with 100 examples each; for each bootstrap sample, we learnt a model; and finally, we averaged the models by selecting the mean pT value among the 1000 ones we fit. In the machine-learning literature, this technique is known as bootstrap aggregating, or bagging (Breiman, 1996).

To test the aggregated model, we implemented RoTS-MCP, a robust tabu search (Taillard, 1991), which sets a value for the tabu list length at each step, selecting it randomly in the interval $[\mathrm{pT} - 10, \mathrm{pT} + 10]$. Since we implemented a leave-one-out cross-validation, the results obtained are an unbiased estimator of the performance of RoTS-MCP.

Table 5 compares the performance of RTS-MCP with RoTS-MCP. The median steps to reach the best-known solutions are comparable with the notable exception of the hard C4000.5 instance and the Brockington–Culberson family of instances. In the easier gen400_p0.9_55 and p-hat300-2 instances, RTS-MCP finds the best-known solutions with a smaller number of steps than RoTS-MCP, while in other 18 instances the opposite is true.

Overall, the results show that even without knowing *a priori* the best fixed parameter values for the instance at hand, it is possible to predict a parameter setting that achieves results that are competitive with a state-of-the-art algorithm by measuring instance properties. The difference between the two algorithms across all instances is small. A stratified one-sided rank-based permutation test (akin to a stratified version of the Mann–Whitney U-test) at a 0.05 significance level rejects the null hypothesis in favour of RoTS-MCP finding best-known solutions in fewer steps than RTS-MCP. The test is implemented in the R coin package (Hothorn et al., 2008).

3. The quadratic assignment problem

To understand if the results on RTS-MCP are representative of a more general picture, we apply the same type of analysis to RTS-QAP, which is presented in Section 2. RTS-QAP is the first RTS

Table 5
Comparison of the median steps of RTS-MCP and RoTS-MCP

Instance	RTS-MCP	RoTS-MCP
C125.9	84.0	**70.0**
C250.9	1147.0	**914.0**
C500.9	81,144.0	**36,677.0**
C1000.9	708,530.0	**336,699.5**
C2000.9	*0.4%*	***0.9%***
DSJC1000.5	34,560.0	33,761.0
DSJC500.5	1400.5	**1235.5**
C2000.5	32,772.0	**26,371.0**
C4000.5	**3,677,632.0**	4,660,693.0
MANN_a27	75,262.0	75,223.5
MANN_a45	*0.2%*	***0.3%***
MANN_a81	***0.2%***	*0.0%*
brock200_2	**56,583.0**	137,126.5
brock200_4	**178,136.0**	332,130.0
brock400_2	***93.4%***	*76.4%*
brock400_4	**1,699,627.0**	2,908,981.5
brock800_2	***1.0%***	*0.5%*
brock800_4	***14.5%***	*11.5%*
gen200_p0.9_44	1429.0	**1108.0**
gen200_p0.9_55	584.0	**485.0**
gen400_p0.9_55	**21,150.5**	23,553.0
gen400_p0.9_65	1390.0	**1219.0**
gen400_p0.9_75	1583.0	**1303.0**
hamming10-4	491.0	571.0
keller5	3040.0	**1843.0**
keller6	672,768.0	**329,380.5**
p_hat300-1	139.0	**96.0**
p_hat300-2	**27.0**	33.0
p_hat300-3	620.0	**469.0**
p_hat700-1	1182.0	**974.0**
p_hat700-2	118.0	**94.0**
p_hat700-3	210.0	216.0
p_hat1500-1	139,617.0	**98,601.0**
p_hat1500-2	363.0	265.0
p_hat1500-3	1229.0	**733.0**

Note: Statistically significant improvements are highlighted in boldface. Instances highlighted in italics are those which one of the two algorithms was not able to solve with 100% success rate.

algorithm proposed in the literature (Battiti and Tecchiolli, 1994), and a state-of-the-art algorithm for some instance classes.

The QAP is to find a minimal cost assignment between a set of facilities P and a set of locations L that minimises a quadratic cost function. Let W and D be two square matrices; the first represents a set of weights or flows between facilities, defined by the weight function $w : P \times P \to \mathbb{R}$; the second represents a set of distances between locations, defined by the distance function $d : L \times L \to \mathbb{R}$. The problem is to find a bijective function $f : P \to L$ that assigns each facility to a location and

minimises the cost functional:

$$\sum_{i,j \in P} w_{i,j} d_{\pi(i),\pi(j)},$$

where π is a permutation. In fact, since the number of facilities is equal to the number of locations in the QAP, a permutation π can be used to represent a valid assignment.

3.1. RTS-QAP

RTS-QAP uses a reactive mechanism that operates on three parameters. The algorithm starts with a random permutation that represents a valid assignment. RTS-QAP uses the two-exchange neighbourhood, where to a given permutation π, all permutations π' are neighbours that can be obtained by exchanging two positions in the permutation. At every step, RTS-QAP selects the best possible move in the neighbourhood, and ensures diversification by means of a tabu search with aspiration criterion and perturbations triggered by the reactive mechanism. The aspiration criterion allows two exchanges that improve the quality of the best solution found so far even if they are tabu. The reactive mechanism works as follows. At every step, the current solution is stored in a data structure that contains the history of the search. Every time a solution is re-encountered during the search, RTS-QAP increases the tabu list length by a factor 1.1. We refer as cycle length the number of steps between two successive visits to the same solution. The algorithm keeps track of an exponential moving average of the cycle lengths, which serves two purposes. First, if the number of steps since the last update of the tabu list length is greater than the moving average, the tabu list length is decreased by a factor 0.9. Second, if the algorithm visits the same repeated solution three times in a row, it assumes that it is trapped in the attraction basin of a local minimum. In this case, the current solution is perturbed by a number of random two exchanges that are proportional to the moving average of the cycle length. Overall, the search history and reactive adaptation impact on three parameters of the algorithm: the tabu list length, the perturbation frequency and the perturbation size.

Before starting with the analysis, some details should to be mentioned. There are actually two versions of RTS-QAP, which differ in the way the solutions are hashed and stored in the search history. The first version, $\text{RTS-QAP}_{\text{f}}$, stores the objective function value as a key of the solution. The second version, $\text{RTS-QAP}_{\text{conf}}$, stores a key computed by accumulating the values in the permutation representing the solution with successive shift and xor bit-operations. The first version requires no extra computation and is therefore faster. However, some instance families are characterised by many solutions sharing the same objective function value. For such instance families, computing a key on the permutation representing the solution is necessary to avoid too many false-positives that bias the search towards an unnecessary diversification. On other problem instances, $\text{RTS-QAP}_{\text{f}}$ is the better option. In the rest of the paper, we present the results of $\text{RTS-QAP}_{\text{best}}$, which is the better performing between $\text{RTS-QAP}_{\text{f}}$ and $\text{RTS-QAP}_{\text{conf}}$ for the instance family at hand.

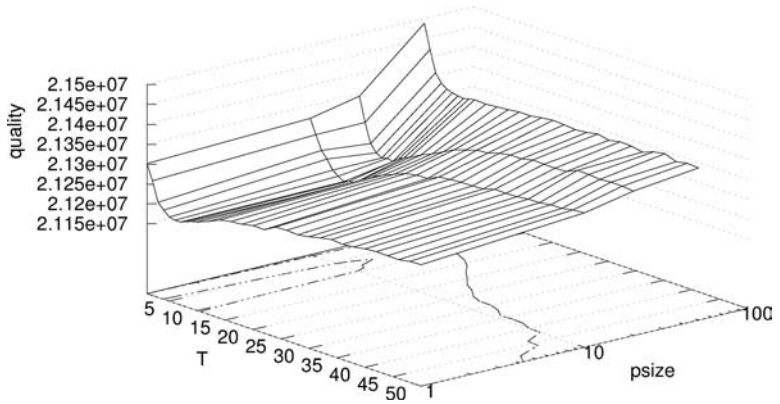


Fig. 4. Median solution quality across 100 runs of $\mathsf{Fixed}_{\mathsf{TLP}}$-QAP on instance tai100a.

3.2. The benchmark set

QAPLIB is the standard benchmark set for comparing heuristics for the QAP (Burkard et al., 1997; http://www.seas.upenn.edu/qaplib). We selected 19 instances with more than 20 facilities from three well-known families: (a) Taillard's uniform randomly generated instances (Taillard-a); (b) Taillard's structured asymmetric randomly generated instances (Taillard-b); (c) Skorin–Kapov instances that have grid-based distances and random weights in the flow matrices.

We begin our study by analysing the impact on the average solution quality of the algorithm parameters that are adapted by reactive search. For this, we implemented $\mathsf{Fixed}_{\mathsf{TLP}}$-QAP, an algorithm identical to RTS-QAP, but with fixed parameter settings. In $\mathsf{Fixed}_{\mathsf{TLP}}$-QAP, the tabu list length T and the perturbation size psize are the same parameters adapted in RTS-QAP; while the perturbations are triggered by a parameter (pert) that controls the number of non-improving steps before a restart is applied. We will refer to pert as perturbation rate. In this section, we present three plots that are representative of the three instance families in the benchmark set. In each plot, we fix pert to 50, and plot the average solution quality against fixed values of the tabu list length and the perturbation size.

Each point, in Figs. 4–6, is the median solution quality over 100 runs of 60 CPU-seconds on the reference machine. The surfaces are much flatter compared to those shown for RTS-MCP, since the variability in solution quality is generally much smaller than in computation time. Still, a clear pattern emerges on the whole benchmark set. On the Taillard-a family of instances (Fig. 4), a good tabu list length is crucial for achieving good-quality solutions, while on the Taillard-b and Skorin–Kapov family of instances (Figs. 5 and 6), spotting the right perturbation size is much more important than spotting the right tabu list length value. In Fig. 5, the horizontal axes are flipped to give a better view of the surface.

3.3. Fixed and random parameter settings

As for RTS-MCP, we study RTS-QAP's ability to adapt parameter settings to local characteristics of the search space. As before, we compare RTS-QAP with $\mathsf{Fixed}_{\mathsf{TLP}}$-QAP, that is, the same algorithm

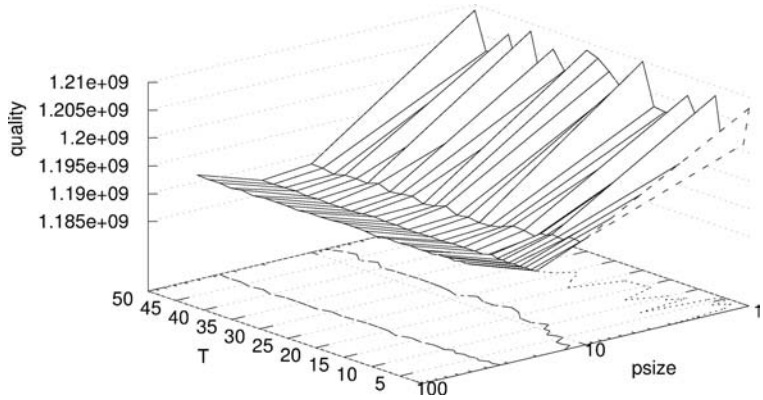


Fig. 5. Median solution quality across 100 runs of $\mathsf{Fixed}_{\mathsf{TLP}}$-QAP on instance tai100b.

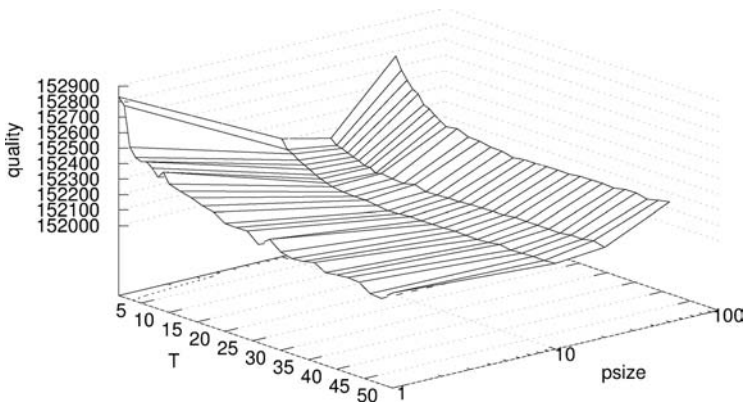


Fig. 6. Median solution quality across 100 runs of $\mathsf{Fixed}_{\mathsf{TLP}}$-QAP on instance sko100a.

but with the tabu list length, perturbation size and perturbation rate parameters fixed to the best values found during the study presented in Section 3.2.

Table 6 shows the median solution quality of $\mathsf{RTS\text{-}QAP}_{\mathsf{best}}$ and $\mathsf{Fixed}_{\mathsf{TLP}}$-QAP over 1000 runs of 60 CPU-seconds on the reference machine. From this table, it is clear that, for almost all instances, the fixed settings of the parameters lead to better median solution qualities. For the cases highlighted in boldface, the improvement of $\mathsf{Fixed}_{\mathsf{TLP}}$-QAP over $\mathsf{RTS\text{-}QAP}_{\mathsf{best}}$ is statistically significant as assessed with a Wilcoxon rank-sum test at 0.05 significance level.

To further test whether RTS-QAP adapts to the local characteristics of the search space, we compare it with $\mathsf{Random}_{\mathsf{ED}}$-QAP, which corresponds to RTS-QAP, but with the parameters set randomly with the same empirical distribution we measured on RTS-QAP. The tabu list length T is updated at every step with a random value, while a new value for pert and psize are generated at random every time a perturbation occurs. In this comparison, we run $\mathsf{RTS\text{-}QAP}_{\mathsf{best}}$ for 60 CPU-seconds on the reference machine, we measure the number of steps performed by $\mathsf{RTS\text{-}QAP}_{\mathsf{best}}$, and we then run $\mathsf{Random}_{\mathsf{ED}}$-QAP for the measured amount of steps. This choice has been made to give a positive bias to $\mathsf{RTS\text{-}QAP}_{\mathsf{best}}$. In fact, $\mathsf{Random}_{\mathsf{ED}}$-QAP has no costly memory operations, and

Table 6
Comparison of the median objective function values of RTS-QAP$_{best}$ and Fixed$_{TLP}$-QAP

Instance	RTS-QAP$_{best}$	Fixed$_{TLP}$-QAP
tai40a.dat	3,146,514.0	**3,143,132.0**
tai50a.dat	4,966,432.0	**4,964,043.0**
tai60a.dat	7,250,862.0	**7,247,540.0**
tai80a.dat	13,600,443.0	**13,598,110.0**
tai100a.dat	21,182,692.0	21,183,094.0
tai40b.dat	637,250,948.0	637,250,948.0
tai50b.dat	459,220,263.0	**458,830,119.0**
tai60b.dat	608,874,816.0	610,393,768.0
tai80b.dat	823,590,370.5	**819,032,289.5**
tai100b.dat	1,189,813,387.0	**1,187,277,747.0**
sko42.dat	15,812.0	15,812.0
sko49.dat	23,386.0	23,386.0
sko56.dat	34,462.0	**34,458.0**
sko64.dat	48,498.0	48,498.0
sko72.dat	66,290.0	**66,272.0**
sko81.dat	91,042.0	**91,022.0**
sko90.dat	115,662.0	**115,598.0**
sko100a.dat	152,140.0	**152,082.0**
sko100b.dat	153,986.0	**153,930.0**
sko100c.dat	147,930.0	**147,881.0**

Note: Statistically significant improvements are highlighted in boldface.

it could therefore perform more steps in the same amount of CPU-time. Table 7 shows that on all instances, a random setting leads to median solution qualities over 1000 runs that are close to the one obtained by RTS-QAP$_{best}$. The overall conclusions that we can derive is that there is no evidence that the algorithm is adapting to the local characteristic of the search space. Only for the Skorin–Kapov family of instances, the solutions qualities are in most cases slightly better for RTS-QAP$_{best}$.

3.4. A robust tabu search

In the previous section, we were able to improve the median solution quality by fixing the parameter to the best values from the analysis on the instances in the benchmark set. At this point, as for the MCP, we are interested in inferring those best fixed values from instance properties that can be measured quickly at the beginning of the search. Again, the aim here is not to improve the state of the art but to see if it is possible to infer the best fixed values without the reactive mechanism.

We measured the size of the instance, dominance, sparsity and skewness of both the distance and flow matrices in which an instance is encoded. For dealing effectively with all these attributes, we resorted to support vector machines for regression (Joachims, 1999) and learnt a model for the tabu list length, perturbation size and perturbation rate, by means of a leave-one-out cross-validation. The model for the tabu list length is a polynomial of second degree, while the models for the perturbation size and the parameter that triggers the perturbations are linear models.

Table 7
Comparison of the median objective function values of RTS-QAP$_{best}$ and Random$_{ED}$-QAP

Instance	RTS-QAP$_{best}$	Random$_{ED}$-QAP
tai40a.dat	3,146,514.0	3,146,258.0
tai50a.dat	4,966,432.0	4,966,070.0
tai60a.dat	7,250,862.0	**7,249,837.0**
tai80a.dat	13,600,443.0	**13,598,473.0**
tai100a.dat	21,182,692.0	**21,179,800.0**
tai40b.dat	637,250,948.0	637,250,948.0
tai50b.dat	459,220,263.0	**459,151,036.0**
tai60b.dat	608,874,816.0	608,863,860.0
tai80b.dat	823,590,370.5	823, 994,642.5
tai100b.dat	1,189,813,387.0	1,189,648,641.5
sko49.dat	23,386.0	23,386.0
sko56.dat	34,462.0	34,462.0
sko64.dat	48,498.0	48,498.0
sko72.dat	**66,290.0**	66,298.0
sko81.dat	**91,042.0**	91,054.0
sko90.dat	**115,662.0**	115,670.0
sko100a.dat	**152,140.0**	152,154.0
sko100b.dat	**153,986.0**	153,994.0
sko100c.dat	**147,930.0**	147,942.0

Note: Statistically significant improvements are highlighted in boldface.

Table 8
Comparison of the median objective function values of RTS-QAP$_{best}$ and RoTS$_{SVM\text{-}QAP}$

Instance	RTS-QAP$_{best}$	RoTS$_{SVM\text{-}QAP}$
tai40a.dat	3,146,514.0	**3,143,132.0**
tai50a.dat	4,966,432.0	**4,965,682.0**
tai60a.dat	7,250,862.0	**7,247,344.0**
tai80a.dat	13,600,443.0	**13,596,785.0**
tai100a.dat	21,182,692.0	21,183,809.0
tai40b.dat	637,250,948.0	637,250,948.0
tai50b.dat	459,220,263.0	**458,870,273.0**
tai60b.dat	608,874,816.0	**608,421,396.0**
tai80b.dat	823,590,370.5	**818,907,741.5**
tai100b.dat	1,189,813,387.0	**1,187,741,740.5**
sko49.dat	23,386.0	23,386.0
sko56.dat	34,462.0	**34,458.0**
sko64.dat	48,498.0	48,498.0
sko72.dat	66,290.0	**66,274.0**
sko81.dat	91,042.0	**91,028.0**
sko90.dat	115,662.0	**115,618.0**
sko100a.dat	152,140.0	**152,090.0**
sko100b.dat	153,986.0	**153,940.0**
sko100c.dat	147,930.0	**147,888.0**

Note: Statistically significant improvements are highlighted in boldface.

As with RoTS-MCP, to construct the data used for the regression, for each instance we order the parameter setting by the median solution quality achieved by $\mathsf{Fixed}_{\mathsf{TLP}}$-QAP, and consider the first 10% for the data. Also in this case, for the model, we resorted to bagging: we randomly selected with replacement 1000 bootstrap samples with 100 examples each, and learnt the three models independently. To preserve possible correlations between the parameters, we consider the models learnt on the bootstrap sample i as a vector $\mathbf{m}_i = (T_i, \mathsf{psize}_i, \mathsf{pert}_i)$, where T_i, psize_i and pert_i are the normalised values obtained from the models. When aggregating the models we select the mean vector $\mathbf{m}_k$, that is, the vector with smallest Euclidean distance from all other vectors:

$$\mathbf{m}_k = \arg\min_{\mathbf{m}_k} \sum_{j \neq k} \|\mathbf{m}_k - \mathbf{m}_j\|.$$

We implemented $\mathsf{RoTS}_{\mathsf{SVM}}$-QAP, a robust tabu search (Taillard, 1991), which is identical to $\mathsf{Fixed}_{\mathsf{TLP}}$-QAP except for the adaptation of the three parameters. At each step of the algorithm, the tabu list length T is set randomly allowing a maximum deviation of ± 5 around the value predicted by the aggregated model. The same is done for the other two parameters: the deviations allowed around the values predicted by the aggregated models are of ± 5 for psize and ± 10 for pert. The deviations have been set in an *ad hoc* way, by looking at the variability of the parameter settings in the data.

Table 8 shows the median solution qualities of the two algorithms over 1000 runs of 60 CPU-seconds on the reference machine. The solution qualities achieved by $\mathsf{RoTS}_{\mathsf{SVM}}$-QAP are significantly better than $\mathsf{RTS\text{-}QAP}_{\mathsf{best}}$ in 15 of 19 instances as assessed by a Wilcoxon rank-sum test at significance level 0.05. A stratified one-sided rank-based permutation test (akin to a stratified version of the Mann–Whitney U-test) rejects, at a 0.05 significance level, the null hypothesis in favour of $\mathsf{RoTS}_{\mathsf{SVM}}$-QAP, finding solutions with better solution quality than $\mathsf{RTS\text{-}QAP}_{\mathsf{best}}$.

4. Conclusions

Our aim in this work was to examine two hypotheses on the parameter adaptation scheme of RTS. The first hypothesis we studied was that the adaptation scheme converges to an instance-specific value that is close to the best value for the instance being optimised. We began our study on RTS-MCP by analysing the benchmark instances used in the literature. Based on the surface of the number of steps to reach the best-known solution for combinations of the two parameters—the tabu list length and restart parameter that drive diversification—it is clear that for most instances, there is a large interval of good values for the tabu list length that allows to find the best-known solution quickly. For the Mannino family of instances, the opposite is true, the restart parameter is important to converge quickly to the best-known solutions, while the tabu list length plays a less important role. By studying the dynamics of the tabu list length adaptation, we observed that RTS-MCP's adaptation scheme converges within few steps to good, instance-wise values of the tabu list length with average values slightly larger or smaller than the instance-optimal tabu list length.

As to the second hypothesis, we observed that the parameter adaptation scheme implemented by RTS does not adapt to the local characteristics of the search space, or if it adapts, the adaptation is not effective. We drew these conclusions after comparing the average number of steps to find the

best-known solution of RTS-MCP with an algorithm that uses the best fixed tabu list length for the instance being optimised. An effective adaptation of the parameter setting to the local characteristics of the search space should give RTS-MCP an advantage over an analogous algorithm that keeps the parameter setting fixed to the best instance-wise value. However, using a fixed parameter setting improves over RTS-MCP in almost all instances. Moreover, setting the parameter randomly with the same empirical distribution of the reactive adaptation scheme leads to results that are close to the results obtained from RTS-MCP. Even a uniformly random parameter setting is competitive, which suggests that the impact of the adaptation scheme on the algorithm ability to find quickly best-known solutions is fairly limited.

To see whether these results could be representative of a more general picture, we extended the study on RTS-QAP. Also in this case, we found that a fixed parameter setting improves the solution qualities achieved by the reactive algorithm. Furthermore, setting the parameters with the same empirical distribution of the parameter adaptation scheme leads to solution qualities that are close to the solution qualities obtained from RTS-QAP. This confirms that, also in this case, there is no evidence that the algorithm effectively adapts to local characteristics of the search space.

Eventually, we also showed that the results obtained by RTS-MCP and RTS-QAP can be matched and for some instances even improved, with a RoTS, in which the parameters are set randomly with small deviations from settings that are modelled from instance properties. Single models have been fit on the whole benchmark sets, but it is reasonable to expect even better results by learning different settings for specific instance families. These experiments were not conducted with the goal of improving the state of the art, but to show that if RTS is able to converge to instance-specific parameter settings, there are other simple alternatives that are able to achieve same results.

RTS is extremely effective across a heterogeneous set of instance families, with no need for tuning the parameters off-line and small sensitivity to its meta-parameters (Pellegrini et al., 2011). Nevertheless, it is interesting to look into the details to understand which aspect contributes more to its efficacy, and in this work, we shed some light in this direction. Further investigation is required to understand why the reactive parameter adaptation studied is not able to exploit the local characteristics of the search space. For example, the reason could be that it reacts too slowly to have measurable effects on the algorithm performance. A further natural extension of this work could be aimed at analysing further reactive search algorithms, and more generally to other parameter adaptation schemes.

Acknowledgements

This work was supported by the META-X project, an *Action de Recherche Concertée* funded by the Scientific Research Directorate of the French Community of Belgium. Franco Mascia, Mauro Birattari and Thomas Stützle acknowledge support from the Belgian F.R.S.-FNRS, of which they are Post-doctoral Researcher and Research Associates. Paola Pellegrini has carried out part of the study reported in this paper while working at IRIDIA, CoDE, Université Libre de Bruxelles, funded by a Bourse d'excellence Wallonie-Bruxelles International.

References

Adamcsek, B., Palla, G., Farkas, I.J., Derényi, I., Vicsek, T., 2006. Cfinder: locating cliques and overlapping modules in biological networks. *Bioinformatics* 22, 8, 1021–1023.

Balas, E., Yu, C., 1986. Finding a maximum clique in an arbitary graph. *SIAM Journal of Computing* 15, 4, 1054–1068.

Bastos, M.P., Ribeiro, C.C., 2001. Reactive tabu search with path-relinking for the Steiner problem in graphs. In Ribeiro, C.C. and Hansen, P. (eds) *Essays and Surveys in Metaheuristics*, Kluwer Academic Publishers, Norwell, MA, pp. 39–58.

Battiti, R., Bertossi, A.A., 1999. Greedy, prohibition, and reactive heuristics for graph partitioning. *IEEE Transactions on Computers* 48, 4, 361–385.

Battiti, R., Brunato, M., Mascia, F., 2008. Reactive search and intelligent optimization. *Operations Research/ Computer Science Interfaces*, Vol. 45. Springer Verlag, New York.

Battiti, R., Mascia, F., 2010. Reactive and dynamic local search for max-clique: engineering effective building blocks. *Computers & Operations Research* 37, 3, 534–542.

Battiti, R., Protasi, M., 2001. Reactive local search for the maximum clique problem. *Algorithmica* 29, 4, 610–637.

Battiti, R., Tecchiolli, G., 1994. The reactive tabu search. *ORSA Journal on Computing* 6, 126–140.

Boginski, V., Butenko, S., Pardalos, P.M., 2006. Mining market data: a network approach. *Computers & Operations Research* 33, 11, 3171–3184 (Special issue: *Operations Research and Data Mining*).

Bomze, I.M., Budinich, M., Pardalos, P.M., Pelillo, M., 1999. The maximum clique problem. *Handbook of Combinatorial Optimization,* Supplement Vol. A, 4, 1–74. Springer, New York.

Breiman, L., 1996. Bagging predictors. *Machine Learning* 24, 123–140.

Brockington, M., Culberson, J.C., 1996. Camouflaging independent sets in quasi-random graphs.*Cliques, Coloring and Satisfiability: Second DIMACS Implementation Challenge*, Vol. 26, *DIMACS Series*. American Mathematical Society, Providence, RI.

Burkard, R.E., Karisch, S.E., Rendl, F., 1997. QAPLIB—a quadratic assignment problem library. *Journal of Global Optimization* 10, 391–403.

Butenko, S., Wilhelm, W.E., 2005. Clique-detection models in computational biochemistry and genomics. *European Journal of Operational Research* 173, 1–17.

Chiang, W.-C.C., Russell, R.A., 1997. A reactive tabu search metaheuristic for the vehicle routing problem with time windows. *INFORMS Journal on Computing* 9, 417–430.

Dorigo, M., Maniezzo, V., Colorni, A., 1991. The ant system: an autocatalytic optimizing process. Technical Report 91-016 Revised, Dipartimento di Elettronica, Politecnico di Milano, Italy.

Dorigo, M., Stützle, T., 2004. *Ant Colony Optimization*. MIT Press, Cambridge, MA.

Eiben, Á.E., Michalewicz, Z., Schoenauer, M., Smith, J.E., 2007. Parameter control in evolutionary algorithms. In Lobo, F., Lima, C.F. and Michalewicz, Z. (eds) *Parameter Setting in Evolutionary Algorithms*. Springer, Berlin, pp. 19–46.

Glover, F., 1986. Future paths for integer programming and links to artificial intelligence. *Computers & Operations Research* 13, 5, 533–549.

Grosso, A., Locatelli, M., Pullan, W., 2007. Simple ingredients leading to very efficient heuristics for the maximum clique problem. *Journal of Heuristics* 14, 6, 587–612.

Hansen, P., Jaumard, B., 1990. Algorithms for the maximum satisfiability problem. *Computing* 44, 279–303.

Hothorn, T., Hornik, K., van de Wiel, M.A., Zeileis, A., 2008. Implementing a class of permutation tests: the coin package. *Journal of Statistical Software* 28, 8, 1–23.

Ji, Y., Xu, X., Stormo, G.D., 2004. A graph theoretical approach to predict common RNA secondary structure motifs including pseudoknots in unaligned sequences. *Bioinformatics* 20, 10, 1591–1602.

Joachims, T., 1999. Making large-scale SVM learning practical. In Schölkopf, B., Burges, C.J.C., Smola, A.J. (eds) *Advances in Kernel Methods—Support Vector Learning*. MIT Press, Cambridge, MA, pp. 169–184.

Johnson, D.J., Trick, M.A. (eds), 1996. Cliques, coloring, and satisfiability: second DIMACS implementation challenge. Workshop, October 11–13, 1993. American Mathematical Society, Boston, MA.

Lourenço, H.R., Martin, O., Stützle, T., 2010. Iterated local search: framework and applications. In Gendreau, M. and Potvin, J.-Y. (eds) *Handbook of Metaheuristics* (2nd edn) Vol. 146, *International Series in Operations Research & Management Science*. Springer, New York, pp. 363–397.

Mascia, F., Cilia, E., Brunato, M., Passerini, A., 2010. Predicting structural and functional sites in proteins by searching for maximum-weight cliques. In Fox, M. and Poole, D. (eds), *Proceedings of the 24th AAAI Conference on Artificial Intelligence, AAAI 2010*, AAAI Press, Menlo Park, CA, pp. 1274–1279.

Mascia, F., Pellegrini, P., Stützle, T., Birattari, M., 2011. An analysis of parameter adaptation in reactive tabu search. Available at http://iridia.ulb.ac.be/supp/IridiaSupp2011-027/. IRIDIA Supplementary page.

Moscato, P., 1999. *Memetic Algorithms: A Short Introduction*. McGraw-Hill, Maidenhead, pp. 219–234.

Nanry, W.P., Wesley Barnes, J., 2000. Solving the pickup and delivery problem with time windows using reactive tabu search. *Transportation Research Part B* 34, 2, 107–121.

Osman, I.H., Wassan, N.A., 2002. A reactive tabu search meta-heuristic for the vehicle routing problem with back-hauls. *Journal of Scheduling* 5, 4, 263–285.

Palla, G., Barabasi, A., Vicsek, T., 2007. Quantifying social group evolution. *Nature* 446, 664–667.

Pellegrini, P., Mascia, F., Stützle, T., Birattari, M., 2011. On the sensitivity of reactive tabu search to its meta-parameters. Technical Report TR/IRIDIA/2011-025, IRIDIA, Université Libre de Bruxelles, Belgium.

Pevzner, P.A., Sze, S.-H., 2000. Combinatorial approaches to finding subtle signals in DNA sequences. *Proceedings of the 8th International Conference on Intelligent Systems for Molecular Biology*. AAAI Press, Menlo Park, CA, pp. 269–278.

Pla, F., Marchant, J.A., 1997. Matching feature points in image sequences through a region-based method. *Computer Vision and Image Understanding* 66, 3, 271–285.

Pullan, W., 2006 Phased local search for the maximum clique problem. *Journal of Combinatorial Optimization* 12, 3, 303–323.

Pullan, W., Hoos, H.H., 2006. Dynamic local search for the maximum clique problem. *Journal of Artificial Intelligence Research* 25, 1, 159–185.

Pullan, W., Mascia, F., Brunato, M., 2011. Cooperating local search for the maximum clique problem. *Journal of Heuristics* 17, 2, 181–199.

Ryan, J.L., Bailey, T.G., Moore, J.T., Carlton, W.B., 1998. Reactive tabu search in unmanned aerial reconnaissance simulations. *Proceedings of the 30th Conference on Winter Simulation*. IEEE Computer Society Press, Los Alamitos, CA, pp. 873–880.

Stützle, T., López-Ibáñez, M., Pellegrini, P., Maur, M., Montes de Oca, M. A., Birattari, M., Dorigo, M., 2012. Parameter adaptation in ant colony optimization. In Hamadi, Y., Monfroy, E., and Saubion, F. (eds) *Autonomous Search*. Springer, Berlin, pp. 191–215.

Taillard, É., 1991. Robust taboo search for the quadratic assignment problem. *Parallel Computing* 17, 4–5, 443–455.

Toune, S., Fudo, H., Genji, T., Fukuyama, Y., Nakanishi, Y., 1998. A reactive tabu search for service restoration in electric power distribution systems. *Proceedings on IEEE World Congress on Computational Intelligence—Evolutionary Computation*. IEEE, Piscataway, NJ, pp. 763–768.

Voy, B.H., Scharff, J.A., Perkins, A.D., Saxton, A.M., Borate, B., Chesler, E.J., Branstetter, L.K., Langston, M.A., 2006. Extracting gene networks for low-dose radiation using graph theoretical algorithms. *PLoS Computational Biology* 2, 7, e89.

Appendix

Solution of size 1100 for MANN_a81

2, 3, 4, 6, 20, 21, 25, 26, 31, 44, 46, 47, 49, 53, 55, 57, 60, 61, 69, 79, 82, 86, 88, 91, 94, 99, 102, 105, 107, 109, 113, 117, 118, 123, 125, 127, 131, 134, 137, 141, 143, 145, 150, 152, 154, 159, 162, 164, 167, 170, 173, 176, 179, 181, 185, 188, 190, 195, 197, 199, 203, 206, 209, 212, 215, 217, 221, 224, 226, 230, 232, 235, 240, 243, 246, 249, 252, 253, 258, 261, 263, 267, 268, 272, 275, 279, 281, 284, 286, 290, 292, 296, 300, 302, 304, 307, 312, 314, 317, 320, 322, 327, 328, 332, 335, 339, 341, 345, 348, 351, 354, 357, 360, 362, 366, 369, 372, 375, 377, 381, 383, 386, 389, 392, 395, 399, 401, 404, 406, 410, 413, 417, 420, 423, 424, 428, 431, 435, 438, 440, 443, 446, 450, 452, 456, 458, 460, 464, 467, 469, 474, 475, 478,

481, 484, 488, 491, 494, 496, 500, 503, 507, 509, 513, 516, 518, 522, 524, 526, 529, 534, 536, 538, 543, 545, 549, 551, 554, 558, 560, 562, 566, 568, 572, 575, 577, 581, 583, 586, 589, 592, 596, 598, 602, 605, 609, 610, 614, 616, 619, 622, 625, 630, 631, 635, 637, 641, 643, 646, 649, 652, 655, 658, 661, 665, 667, 672, 673, 677, 679, 682, 686, 688, 692, 695, 697, 700, 704, 706, 711, 713, 715, 718, 721, 724, 727, 731, 733, 736, 740, 742, 746, 748, 751, 755, 758, 761, 763, 767, 769, 772, 776, 778, 783, 785, 787, 792, 795, 798, 800, 802, 806, 809, 812, 815, 819, 820, 823, 826, 829, 832, 835, 838, 842, 846, 847, 850, 855, 856, 859, 864, 865, 868, 872, 874, 878, 881, 883, 887, 891, 893, 895, 900, 903, 906, 909, 910, 915, 917, 920, 924, 927, 930, 931, 936, 939, 942, 945, 947, 951, 954, 957, 960, 962, 965, 967, 972, 974, 977, 979, 983, 985, 988, 992, 995, 999, 1002, 1005, 1007, 1011, 1014, 1015, 1019, 1021, 1025, 1027, 1032, 1034, 1037, 1041, 1043, 1046, 1049, 1053, 1055, 1059, 1062, 1064, 1068, 1069, 1074, 1077, 1078, 1083, 1086, 1089, 1092, 1094, 1098, 1101, 1104, 1105, 1109, 1112, 1116, 1117, 1122, 1125, 1127, 1131, 1134, 1135, 1139, 1142, 1145, 1148, 1152, 1154, 1158, 1160, 1162, 1166, 1169, 1171, 1175, 1179, 1180, 1184, 1188, 1190, 1192, 1197, 1200, 1202, 1205, 1207, 1210, 1215, 1218, 1221, 1224, 1226, 1230, 1232, 1234, 1239, 1242, 1243, 1247, 1251, 1254, 1255, 1260, 1262, 1266, 1268, 1270, 1275, 1278, 1279, 1284, 1287, 1290, 1291, 1296, 1298, 1302, 1305, 1308, 1311, 1314, 1316, 1318, 1323, 1325, 1328, 1332, 1334, 1338, 1339, 1343, 1347, 1350, 1353, 1356, 1357, 1362, 1365, 1368, 1371, 1374, 1376, 1380, 1383, 1385, 1388, 1392, 1395, 1397, 1400, 1404, 1406, 1410, 1413, 1415, 1419, 1422, 1425, 1428, 1429, 1434, 1437, 1440, 1442, 1444, 1448, 1452, 1455, 1457, 1461, 1464, 1467, 1470, 1473, 1475, 1478, 1482, 1485, 1486, 1490, 1493, 1495, 1499, 1503, 1505, 1509, 1512, 1515, 1517, 1520, 1524, 1526, 1528, 1531, 1534, 1539, 1541, 1545, 1547, 1549, 1552, 1555, 1560, 1563, 1566, 1569, 1570, 1575, 1578, 1581, 1584, 1587, 1590, 1592, 1594, 1598, 1601, 1603, 1606, 1610, 1613, 1617, 1619, 1621, 1626, 1628, 1630, 1634, 1638, 1639, 1642, 1647, 1650, 1653, 1656, 1657, 1661, 1665, 1668, 1669, 1674, 1677, 1680, 1683, 1686, 1689, 1690, 1695, 1698, 1700, 1704, 1705, 1710, 1711, 1716, 1719, 1722, 1724, 1728, 1731, 1732, 1736, 1740, 1743, 1746, 1749, 1752, 1755, 1758, 1759, 1762, 1767, 1770, 1772, 1776, 1779, 1781, 1785, 1786, 1791, 1792, 1796, 1799, 1803, 1804, 1808, 1812, 1815, 1817, 1821, 1822, 1826, 1830, 1831, 1835, 1838, 1840, 1845, 1846, 1850, 1854, 1856, 1859, 1862, 1865, 1867, 1871, 1874, 1876, 1880, 1883, 1886, 1889, 1891, 1894, 1899, 1901, 1904, 1906, 1911, 1912, 1917, 1920, 1921, 1926, 1927, 1932, 1933, 1936, 1939, 1942, 1946, 1948, 1953, 1956, 1957, 1962, 1963, 1968, 1970, 1973, 1977, 1978, 1983, 1986, 1989, 1992, 1994, 1998, 2000, 2004, 2007, 2010, 2013, 2015, 2017, 2021, 2024, 2027, 2030, 2034, 2036, 2040, 2042, 2044, 2048, 2051, 2055, 2057, 2060, 2063, 2066, 2070, 2072, 2074, 2078, 2082, 2084, 2087, 2091, 2093, 2097, 2100, 2102, 2106, 2108, 2111, 2115, 2118, 2120, 2123, 2127, 2129, 2132, 2136, 2139, 2142, 2143, 2147, 2149, 2154, 2156, 2158, 2163, 2166, 2168, 2172, 2174, 2177, 2180, 2183, 2187, 2189, 2193, 2195, 2197, 2201, 2204, 2207, 2210, 2212, 2216, 2219, 2222, 2226, 2228, 2232, 2235, 2238, 2241, 2242, 2245, 2249, 2253, 2256, 2257, 2262, 2265, 2267, 2271, 2274, 2277, 2278, 2281, 2284, 2289, 2292, 2293, 2297, 2301, 2303, 2306, 2308, 2312, 2314, 2317, 2320, 2323, 2326, 2329, 2332, 2337, 2338, 2342, 2344, 2347, 2351, 2353, 2357, 2359, 2362, 2365, 2368, 2373, 2375, 2377, 2380, 2385, 2387, 2391, 2394, 2397, 2400, 2402, 2404, 2407, 2410, 2415, 2416, 2421, 2422, 2427, 2429, 2433, 2434, 2438, 2440, 2443, 2448, 2449, 2452, 2457, 2459, 2461, 2465, 2467, 2471, 2475, 2476, 2479, 2484, 2487, 2488, 2493, 2494, 2498, 2501, 2503, 2506, 2509, 2512, 2517, 2519, 2523, 2524, 2527, 2531, 2533, 2537, 2540, 2544, 2546, 2549, 2552, 2556, 2557, 2562, 2564, 2568, 2571, 2573, 2576, 2578, 2582, 2585, 2588, 2591, 2594, 2596, 2599, 2603, 2606, 2608, 2613, 2614, 2619, 2622, 2625, 2628, 2631, 2634, 2637, 2640, 2643, 2645, 2649, 2650, 2654, 2657, 2660, 2664, 2666, 2669, 2672, 2675, 2678, 2681, 2684, 2686, 2691, 2692, 2695,

2698, 2701, 2705, 2709, 2712, 2713, 2716, 2721, 2723, 2726, 2730, 2733, 2736, 2738, 2742, 2744, 2748, 2750, 2753, 2755, 2759, 2762, 2765, 2768, 2772, 2774, 2778, 2780, 2784, 2786, 2789, 2793, 2795, 2799, 2802, 2805, 2808, 2809, 2813, 2815, 2820, 2822, 2824, 2829, 2831, 2834, 2836, 2840, 2843, 2846, 2849, 2852, 2854, 2858, 2861, 2865, 2868, 2870, 2872, 2876, 2879, 2883, 2885, 2888, 2891, 2895, 2897, 2900, 2903, 2907, 2909, 2911, 2915, 2917, 2921, 2924, 2928, 2929, 2934, 2937, 2940, 2943, 2945, 2947, 2951, 2955, 2958, 2960, 2964, 2965, 2968, 2973, 2976, 2979, 2982, 2985, 2987, 2991, 2994, 2996, 2999, 3003, 3004, 3008, 3011, 3015, 3017, 3020, 3023, 3026, 3029, 3032, 3035, 3039, 3041, 3043, 3046, 3049, 3052, 3055, 3058, 3062, 3064, 3067, 3071, 3073, 3077, 3080, 3083, 3087, 3089, 3093, 3096, 3099, 3101, 3104, 3106, 3110, 3113, 3115, 3118, 3123, 3125, 3129, 3131, 3135, 3138, 3140, 3143, 3145, 3149, 3152, 3155, 3159, 3160, 3163, 3166, 3170, 3172, 3175, 3178, 3182, 3186, 3188, 3191, 3195, 3197, 3201, 3203, 3205, 3209, 3212, 3215, 3218, 3221, 3225, 3226, 3229, 3233, 3237, 3240, 3241, 3245, 3248, 3252, 3254, 3257, 3259, 3263, 3266, 3270, 3273, 3274, 3279, 3281, 3283, 3288, 3289, 3294, 3297, 3300, 3302, 3306, 3309, 3310, 3314, 3318, 3321.