ISSN 1392 – 124X INFORMATION TECHNOLOGY AND CONTROL, 2007, Vol.36, No.4

# DEFINING TABU TENURE FOR THE QUADRATIC ASSIGNMENT PROBLEM

**Alfonsas Misevičius, Armantas Ostreika**

*Kaunas University of Technology, Department of Multimedia Engineering*
*Studentu St. 50, LT–51368 Kaunas, Lithuania*

**Abstract**. Tabu search (TS) algorithms are among the most efficient heuristic techniques in combinatorial optimization. Within these algorithms, it is important that the proper policies for maintaining the tabu tenure (tabu list size) are applied. In this paper, we discuss the mechanisms of defining the tabu tenure for the famous combinatorial optimization problem – the quadratic assignment problem (QAP). Several variants of maintaining the tabu tenure are implemented. They are examined on the instances taken from the QAP instances library – QAPLIB. The results from the experiments demonstrate that the deterministic strategies of defining the tabu tenure are preferable to the corresponding random strategies.

**Keywords:** combinatorial optimization, heuristics, tabu search, tabu tenure, quadratic assignment problem.

## Introduction

The quadratic assignment problem (QAP) is one of the most difficult combinatorial optimization problems. Formally, it can be formulated as follows [11]. Let two matrices $\boldsymbol{A} = (a_{ij})_{n \times n}$ and $\boldsymbol{B} = (b_{kl})_{n \times n}$ and the set $\Pi$ of all possible permutations of $\{1, 2, ..., n\}$ be given. The goal is to find a permutation $\pi = (\pi(1), \pi(2), ..., \pi(n)) \in \Pi$ that minimizes

$$z(\pi) = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} b_{\pi(i)\pi(j)}. \qquad (1)$$

The quadratic assignment problem is NP-hard [17] and cannot be actually solved to optimality for larger $n$ ($n > 36$) [2]. Therefore, heuristic algorithms are commonly used to get near-optimal (or possibly pseudo-optimal) solutions within reasonable computation time. (For a survey of heuristics for the QAP, see, for example, [3, 5, 16].) Among the heuristic approaches, the tabu search (TS) method has been proven to be a powerful technique, in particular, for solving the quadratic assignment problem [1,6,12,13,19]. The tabu search based algorithms seem to be obligatory – in one or another form – if one seeks for high-quality solutions for this challenging problem.

The remaining part of this paper is organized as follows. We start with some preliminary concepts. In Section 2, we discuss several variants of maintaining the tabu tenure for the quadratic assignment problem. The results of the computational experiments with the various variants of the tabu tenure are presented in Section 3. Finally, Section 4 completes the paper with concluding remarks.

## 1. Preliminary concepts

Let $\Theta_2$: $\Pi \to 2^{\Pi}$ be a 2-exchange neighbourhood function that assigns for every $\pi$ the set of neighbouring solutions such that $\Theta_2(\pi) = \{\pi' \mid \pi' \in \Pi, \; |\{i \mid \pi(i) \neq \pi'(i)\}| = 2\}$. The solution $\pi' \in \Theta_2(\pi)$ can be obtained from $\pi$ by a corresponding move $\phi(\pi, i, j)$: $\Pi \times \mathrm{N} \times \mathrm{N} \to \Pi$, which swaps the $i$-th and $j$-th element in the given permutation, i.e. $\pi'(i) = \pi(j)$, $\pi'(j) = \pi(i)$, $i, j = 1, 2, ..., n$. The solution $\pi^{\bullet} \in \Pi$ is said to be locally optimal with respect to the neighbourhood $\Theta_2$ if $z(\pi^{\bullet}) \leq z(\pi')$ for every $\pi' \in \Theta_2(\pi^{\bullet})$.

The tabu search method is based on the neighbourhood search with local-optima avoidance. The central idea is allowing climbing moves when no improving neighbouring solution exists, i.e. a move is allowed even if a new solution is worse than the current one. The reverse moves must be forbidden. This is done by storing the corresponding solution (or some attribute associated with this solution) in a tabu list ($T$). The tabu list keeps information on the last $h = |T|$ solutions (attributes) which have been considered during the search process. The parameter $h$ is called a tabu tenure (tabu list size).

With the QAP, a common way is organizing the tabu list as an $n \times n$ integer matrix $\boldsymbol{T} = (t_{ij})_{n \times n}$. At the beginning, all the entries of $\boldsymbol{T}$ are set to zero. As the algorithm progresses, the entry $t_{ij}$ stores the current iteration number plus the tabu tenure $h$, i.e. the number of the future iteration starting at which the move $\phi(\cdot, i, j)$ may again be performed. So, the move

$\phi(\cdot, i, j)$ is tabu if the value of $t_{ij}$ is equal or greater than the current iteration number.

Formally, TS starts from an initial solution $\pi$ in $\Pi$. The process is then continued in an iterative way – moving from the current solution to a neighbouring one. At each step of the procedure, the neighbourhood $\Theta_2(\pi)$ of the given solution is explored and the non-tabu move to the solution $\pi' \in \Theta_2(\pi)$ that improve most the objective function value is chosen. The best so far solution is recorded. The process stops as soon as a termination criterion is satisfied.

A more thorough discussion on the principles as well as the specific facets of the tabu search (like target analysis, aspiration criterion, long-term-memory, etc.) can be found in [8, 9, 10, 18 (p. 159–165)].

## 2. Variants of maintaining the tabu tenure for the QAP

The main difficulties within the tabu search algorithms are the cycles of the search trajectories and convergence speed. The cycling trajectories (and, consequently, stagnation of the search) take place when the tabu tenure (tabu list size) is too small. On the other hand, the convergence rate of the search process tends to be low when the tabu tenure is too large. In both cases, it might be impossible to find high-quality solutions within reasonable computation time. Therefore, defining the proper (optimal) value of the tabu tenure is one of the key aspects of the tabu search algorithms.

In this section, we will describe some variants of defining the tabu tenure for the quadratic assignment problem. We will focus only on the modifications related to the tabu tenure. The algorithm itself is identical to the combined enhanced tabu search (CETS) algorithm [13], except the modifications actually discussed. For the more details and the template of CETS, see [13].

### 2.1. Fixed tabu tenure

In the simplest case, the tabu tenure is fixed at a constant value. Typically, this value depends on the problem size, $n$, i.e. $h = cn$ (or more precisely, $h = \lceil cn \rceil$), where $c$ is an empirical parameter (for the QAP, $c$ is somewhere between 0.01 and 2.0 for ordinary $n$'s ($n \le 150$)).

By having the fixed value of the tabu tenure, there is a risk that the phenomenon known as "deterministic chaos" will occur [1]. In this case, "short-term" cycles are absent, but the search trajectories are still confined in a limited part of the solution space – this is like having "long-term" (hidden) cycling. One of the possible ways to try to avoid this cycling is to apply variable tabu tenure instead of the fixed one. Below, we briefly describe several variants of using the non-fixed tabu tenure where the tenure is varied in some deterministic or random manner during the search process.

### 2.2. Periodic tabu tenure

The periodic (cyclic) tabu tenure is an example of the deterministic strategy of maintaining the tabu tenure. In this case, the tabu tenure range $[h_{\min}, h_{\max}]$ is considered (here, $h_{\min} = c_1 n$ and $h_{\max} = c_2 n$, $c_1 < c_2$; the appropriate values of $c_1$ and $c_2$ may be determined experimentally). So, at the beginning, $h$ is equal to the minimum value $h_{\min}$; further, $h$ is increased gradually, step by step, until the maximum value $h_{\max}$ is reached; once the maximum value $h_{\max}$ has been achieved (or possibly a new better locally optimal solution has been encountered), the actual value of $h$ is dropped to $h_{\min}$, and so on. In our experimentation, $h_{\min} \in [0.07n, 0.3n]$ and $h_{\max} \in [0.15n, 0.4n]$, depending on the problem size, $n$.

### 2.3. Random tabu tenure

The random (probabilistic) tabu tenure is similar to the periodic one, except that the value of the tenure is picked randomly from the interval $[h_{\min}, h_{\max}]$. During the search process, $h$ is renewed (regenerated) every $2h_{\max}$ iterations (see also the robust tabu search algorithm [19]). In [19], $h_{\min}$ and $h_{\max}$ are fixed at $0.9n$ and $1.1n$; while, in [7], the range is extended to $[0.2n, 1.8n]$. In this work, $h_{\min}$, $h_{\max}$ vary from $0.07n$ to $0.4n$, depending on $n$.

### 2.4. Dynamic periodic tabu tenure: basic variant

We also propose two additional strategies for maintaining the tabu tenure. They are based on dynamic changing of the value of the tabu tenure. In fact, the tabu tenure is individually defined for the particular move $(\phi(\cdot, i, j)$, $i = 1, 2, \ldots, n-1$, $j = i+1, \ldots, n$; that is, the tabu tenure is tuned for each pair of indices $(i, j)$ (see Figure 1). We use the additional matrix $\boldsymbol{H} = (h_{ij})_{n \times n}$ for storing these dynamic values of the tabu tenure. Then, the tabu list is updated according to the following formula:

$$t_{ij} = q + h_{ij},$$

where $t_{ij}$ denotes the corresponding entry of the tabu list, $q$ is the current iteration number, and $h_{ij}$ denotes the current value of the tabu tenure for the move $\phi(\cdot, i, j)$. The values $h_{ij}$ are maintained according to the rule described in Section 2.2.

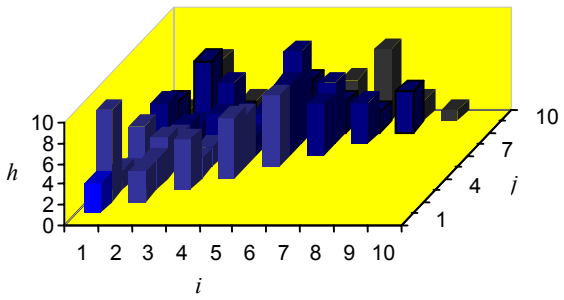


**Figure 1.** Dynamic tabu tenures

The above strategy adds some flexibility and robustness to the search process. It can be viewed as a special type of diversification mechanism.

### 2.5. Dynamic periodic tabu tenure: modified variant

The difference between the basic and modified variant of the dynamic strategy is that, in the modified version, one starts from zero tabu tenure instead of $h_{\min}$. The remaining process is identical to that described in Section 2.4. The illustration of the modified tabu tenure for the separate pair of indices is shown in Figure 2.

The motivation of this modification is based on the assumption that there is no necessity to apply the large tabu tenure at the initial phase of the search (see also Section 2.6).

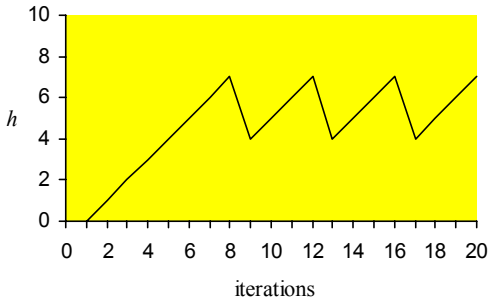


**Figure 2.** Illustration of the modified variant of the tabu tenure

### 2. 6. Using tabu search with delay

It could be conjectured that, in the early stages of the search (provided that the process starts from the statistically-average-quality solution), the cycles are unlikely. For this reason, we do not need to take care of the tabu moves and restrict the exploration of the promising regions of the solution space at the initial iterations of the algorithm. This is what we call the tabu search with delay [13]. In this case, the tabu tenure is equal to zero at the beginning of the search process. After $w$ iterations are performed, the tabu search is continued in the ordinary way. The parameter $w$ is defined as $\lfloor \delta \cdot n \rfloor$, where $\delta$ is a delay factor (we used $\delta = 1$).

## 3. Computational experiments

We have carried out a number of computational experiments in order to test the performance of the different variants of defining the tabu tenure. In the experiments, we used the QAP instances taken from the publicly available QAP library – QAPLIB [4]. We examined the following types of the QAP instances:

a) random instances (these instances are randomly generated according to a uniform distribution; in QAPLIB, they are denoted by tai20a, tai25a, tai30a, tai35a, tai40a, tai50a, tai60a, tai80a, and tai100a);
b) quasi-random instances (in these instances, the values of the matrix $\boldsymbol{A}$ are generated pseudo-randomly and the values of the matrix $\boldsymbol{B}$ are "Manhattan" distances between grid points; the following are the grid-based instances: sko42, sko49, sko56, sko64, sko72, sko81, sko90, sko100a–f, tho30, tho40, tho150, wil50, and wil100);
c) real-life like instances (they are generated in such a way that the entries of the matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ resemble a distribution from real world problems; these instances are denoted by tai20b, tai25b, tai30b, tai35b, tai40b, tai50b, tai60b, tai80b, tai100b, and tai150b).

The variants of the tabu tenure used in the experimentation are as follows: the fixed tabu tenure (FTT), the periodic tabu tenure (PTT), the random tabu tenure (RTT), the basic variant of the dynamic periodic tabu tenure (DPTT-B), and the modified variant of the dynamic periodic tabu tenure (DPTT-M). All these variants have been tried in both the standard (non-delay) mode and the delay mode. The delay variants are denoted by FTT$^d$, PTT$^d$, RTT$^d$, DPTT-B$^d$, and DPTT-M$^d$, respectively. Two sets of the fixed values of the tabu tenure were examined: set 1 ($h = 0.35n$ ($n < 50$), $h = 0.2n$ ($50 \le n < 100$), $h = 0.1n$ ($n \ge 100$)); set 2 ($h = 0.7n$ ($n < 50$), $h = 0.6n$ ($50 \le n < 100$), $h = 0.5n$ ($n \ge 100$)). The corresponding variants are entitled as FTT-1 (FTT-1$^d$) and FTT-2 (FTT-2$^d$). Regarding the non-fixed tenures, we used the following ranges of tabu tenure: a) $h_{\min} = 0.3n$ and $h_{\max} = 0.4n$ ($n < 50$), b) $h_{\min} = 0.15n$ and $h_{\max} = 0.3n$ ($50 \le n < 100$), c) $h_{\min} = 0.07n$ and $h_{\max} = 0.15n$ ($n \ge 100$).

The performance criteria for the algorithms are as follows: a) the average deviation from a (pseudo-)optimal (best known) solution – $\bar{\delta}$ ($\bar{\delta} = 100(\bar{z} - \breve{z})/\breve{z}\,[\%]$, where $\bar{z}$ is the average objective function value over 10 restarts and $\breve{z}$ is the best known value (BKV) of the objective function); b) the number of solutions that are within 1% optimality – $C_{1\%}$; c) the number of the best known solutions – $C_{\text{BKS}}$.

The results of comparison of the different variants of the tabu tenure are presented in Tables 1–9. As can be seen, the computation times are different for the random and real-life like instances. We used relatively larger runs for the random instances and shorter runs for the real-life like instances. This is to get more fairness of the experiments.

From Tables 1–9, it can be observed that the deterministic strategies of maintaining the tabu tenure (in particular, the fixed and periodic tabu tenures) are slightly superior to the random strategy. Surprisingly, the fixed tabu tenure in combination with the delay mechanism yielded the best results (this is especially evident for the quasi-random instances). The reason

may be that the value of the tabu tenure was especially tuned for this type of instances. However, in general it may be risky to use "tweaked" values of the tabu tenure – see the results obtained with the increased tabu tenure (columns "FTT-2"/"FTT-2[d]" in Tables 1–9). On the whole, the tuning of the tabu tenure may be time-consuming. We therefore guess that it is probably better to use the range of tabu tenure instead of the fixed value (however, the adjusting of the range of the tabu tenure is still important).

**Table 1.** Results of comparison of the tabu tenure variants for the random QAP instances (I).
(CPU times per restart are given in seconds. 3 GHz Pentium computer was used in the experiments)

| Instance | $n$ | BKV | $\overline{\delta}$ , $C_{1\%}/C_{BKS}$ | | | | | | | | | | | | CPU time |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | FTT-1 | | FTT-2 | | PTT | | RTT | | DPTT-B | | DPTT-M | | |
| **tai20a** | 20 | 703482 [a] | 0.183 | 10/4 | 0.276 | 10/2 | 0.183 | 10/4 | 0.152 | 10/5 | 0.061 | 10/8 | 0.183 | 10/4 | *0.3* |
| **tai25a** | 25 | 1167256 [a] | 0.254 | 10/4 | 0.189 | 10/5 | 0.203 | 10/5 | 0.180 | 10/5 | 0.320 | 10/1 | 0.168 | 10/5 | *0.7* |
| **tai30a** | 30 | 1818146 [a] | 0.085 | 10/8 | 0.095 | 10/4 | 0.058 | 10/7 | 0.078 | 10/8 | 0.112 | 10/7 | 0.072 | 10/8 | *1.3* |
| **tai35a** | 35 | 2422002 [a] | 0.267 | 10/4 | 0.278 | 10/4 | 0.300 | 10/3 | 0.078 | 10/7 | 0.178 | 10/5 | 0.370 | 10/2 | *2.3* |
| **tai40a** | 40 | 3139370 [a] | 0.478 | 10/0 | 0.590 | 10/0 | 0.398 | 10/0 | 0.430 | 10/0 | 0.460 | 10/0 | 0.454 | 10/0 | *3.8* |
| **tai50a** | 50 | 4941410 [a] | 0.489 | 10/0 | 0.684 | 9/0 | 0.519 | 10/0 | 0.584 | 10/0 | 0.598 | 10/0 | 0.676 | 10/0 | *8.9* |
| **tai60a** | 60 | 7205962 [b] | 0.630 | 10/0 | 0.789 | 10/0 | 0.715 | 10/0 | 0.666 | 10/0 | 0.615 | 10/0 | 0.704 | 10/0 | *18.5* |
| **tai80a** | 80 | 13526696 [c] | 0.525 | 10/0 | 0.651 | 10/0 | 0.571 | 10/0 | 0.480 | 10/0 | 0.545 | 10/0 | 0.523 | 10/0 | *60* |
| **tai100a** | 100 | 21071540 [d] | 0.432 | 10/0 | 0.605 | 10/0 | 0.437 | 10/0 | 0.488 | 10/0 | 0.422 | 10/0 | 0.447 | 10/0 | *147* |

[a] comes from [4]; [b] comes from [12]; [c] comes from [14]; [d] comes from [15].

**Table 2.** Results of comparison of the tabu tenure variants for the quasi-random QAP instances (I).
(CPU times per restart are given in seconds. 3 GHz Pentium computer was used in the experiments)

| Instance | $n$ | BKV | $\overline{\delta}$ , $C_{1\%}/C_{BKS}$ | | | | | | | | | | | | CPU time |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | FTT-1 | | FTT-2 | | PTT | | RTT | | DPTT-B | | DPTT-M | | |
| **sko42** | 42 | 15812 [a] | 0.015 | 10/6 | 0.023 | 10/4 | 0.032 | 10/6 | 0.015 | 10/6 | 0.015 | 10/6 | 0.111 | 10/5 | *1.0* |
| **sko49** | 49 | 23386 [a] | 0.058 | 10/2 | 0.036 | 10/4 | 0.068 | 10/1 | 0.048 | 10/2 | 0.044 | 10/2 | 0.062 | 10/2 | *1.7* |
| **sko56** | 56 | 34458 [a] | 0.196 | 10/2 | 0.237 | 10/3 | 0.158 | 10/2 | 0.282 | 10/0 | 0.152 | 10/1 | 0.245 | 10/1 | *2.5* |
| **sko64** | 64 | 48498 [a] | 0.140 | 10/3 | 0.042 | 10/5 | 0.082 | 10/3 | 0.096 | 10/5 | 0.230 | 10/1 | 0.050 | 10/3 | *4.8* |
| **sko72** | 72 | 66256 [a] | 0.132 | 10/0 | 0.207 | 10/1 | 0.163 | 10/0 | 0.196 | 10/0 | 0.212 | 10/0 | 0.217 | 10/0 | *6.6* |
| **sko81** | 81 | 90998 [a] | 0.285 | 10/0 | 0.157 | 10/0 | 0.220 | 10/0 | 0.177 | 10/0 | 0.220 | 10/0 | 0.210 | 10/0 | *10.0* |
| **sko90** | 90 | 115534 [a] | 0.273 | 10/0 | 0.204 | 10/0 | 0.267 | 10/0 | 0.203 | 10/0 | 0.205 | 10/0 | 0.253 | 10/0 | *13.8* |
| **sko100a** | 100 | 152002 [a] | 0.102 | 10/1 | 0.142 | 10/0 | 0.113 | 10/1 | 0.133 | 10/0 | 0.081 | 10/2 | 0.137 | 10/0 | *32* |
| **sko100b** | 100 | 153890 [a] | 0.197 | 10/1 | 0.148 | 10/2 | 0.193 | 10/0 | 0.132 | 10/1 | 0.196 | 10/0 | 0.183 | 10/0 | *33* |
| **sko100c** | 100 | 147862 [a] | 0.138 | 10/0 | 0.085 | 10/1 | 0.075 | 10/0 | 0.306 | 10/1 | 0.113 | 10/0 | 0.099 | 10/0 | *32* |
| **sko100d** | 100 | 149576 [a] | 0.232 | 10/0 | 0.173 | 10/0 | 0.263 | 10/0 | 0.216 | 10/0 | 0.233 | 10/0 | 0.177 | 10/0 | *33* |
| **sko100e** | 100 | 149150 [a] | 0.344 | 10/0 | 0.145 | 10/0 | 0.216 | 10/1 | 0.280 | 10/0 | 0.325 | 10/0 | 0.257 | 10/1 | *33* |
| **sko100f** | 100 | 149036 [a] | 0.228 | 10/0 | 0.218 | 10/0 | 0.366 | 10/0 | 0.344 | 10/0 | 0.271 | 10/0 | 0.231 | 10/0 | *33* |
| **tho30** | 30 | 149936 [a] | 0.567 | 7/4 | 0.230 | 9/7 | 0.277 | 9/4 | 0.377 | 9/4 | 0.283 | 9/5 | 0.314 | 9/3 | *0.4* |
| **tho40** | 40 | 240516 [a] | 0.149 | 10/0 | 0.154 | 10/0 | 0.118 | 10/0 | 0.120 | 10/0 | 0.183 | 10/1 | 0.041 | 10/0 | *0.9* |
| **tho150** | 150 | 8133398 [a] | 0.274 | 10/0 | 0.239 | 10/0 | 0.259 | 10/0 | 0.214 | 10/0 | 0.299 | 10/0 | 0.245 | 10/0 | *174* |
| **wil50** | 50 | 48816 [a] | 0.056 | 10/0 | 0.054 | 10/1 | 0.045 | 10/1 | 0.073 | 10/0 | 0.065 | 10/2 | 0.073 | 10/1 | *1.9* |
| **wil100** | 100 | 273038 [a] | 0.123 | 10/0 | 0.095 | 10/0 | 0.084 | 10/0 | 0.089 | 10/0 | 0.141 | 10/0 | 0.139 | 10/0 | *33* |

[a] comes from [4].

Unexpectedly, the actual results do not confirm the advantage of the delay modification over the standard variant. The exception is the results for the quasi-random instances obtained by using the fixed and dynamic tenures as well as the results for the random instances obtained by using the dynamic tenures.

**Table 3.** Results of comparison of the tabu tenure variants for the real-life like QAP instances (I).
(CPU times per restart are given in seconds. 3 GHz Pentium computer was used in the experiments)

| Instance | *n* | BKV | $\bar{\delta}$ , $C_{1\%}/C_{BKS}$ | | | | | | | | | | | | CPU time |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | FTT-1 | | FTT-2 | | PTT | | RTT | | DPTT-B | | DPTT-M | | |
| **tai20b** | 20 | 122455319 [a] | 0.181 | 10/6 | 0.181 | 10/6 | 0.226 | 10/5 | 0.221 | 10/5 | 0.136 | 10/7 | 0.136 | 10/7 | *0.1* |
| **tai25b** | 25 | 344355646 [a] | 0.007 | 10/9 | 0.081 | 10/7 | 0.569 | 9/8 | 0.045 | 10/8 | 0.008 | 10/9 | **0** | | *0.2* |
| **tai30b** | 30 | 637117113 [a] | 1.376 | 4/3 | 1.553 | 2/2 | 1.824 | 3/2 | 1.844 | 2/2 | 1.361 | 4/4 | 1.860 | 2/1 | *0.3* |
| **tai35b** | 35 | 283315445 [a] | 0.800 | 8/2 | 1.555 | 6/1 | 1.204 | 6/4 | 1.078 | 6/1 | 0.920 | 8/3 | 0.392 | 8/3 | *0.6* |
| **tai40b** | 40 | 637250948 [a] | 2.156 | 3/3 | 1.989 | 4/4 | 0.894 | 6/6 | 1.464 | 5/5 | 1.693 | 4/4 | 2.510 | 2/2 | *0.9* |
| **tai50b** | 50 | 458821517 [a] | 0.437 | 10/3 | 0.309 | 10/3 | 0.220 | 10/6 | 0.359 | 10/1 | 0.373 | 10/2 | 0.505 | 8/3 | *1.9* |
| **tai60b** | 60 | 608215054 [a] | 0.772 | 10/0 | 0.661 | 6/1 | 0.835 | 5/1 | 1.012 | 4/2 | 0.643 | 6/1 | 0.798 | 5/1 | *4.1* |
| **tai80b** | 80 | 818415043 [a] | 0.759 | 8/0 | 0.771 | 8/0 | 0.938 | 6/0 | 0.991 | 3/0 | 0.820 | 6/0 | 1.007 | 4/0 | *13.0* |
| **tai100b** | 100 | 1185996137 [a] | 0.684 | 8/0 | 1.041 | 6/1 | 0.516 | 9/1 | 0.605 | 8/2 | 0.988 | 7/1 | 0.921 | 7/0 | *33* |
| **tai150b** | 150 | 498896643 [b] | 0.503 | 10/0 | 0.602 | 10/0 | 0.561 | 10/0 | 0.540 | 10/0 | 0.528 | 10/0 | 0.618 | 10/0 | *101* |

[a] comes from [4]; [b] comes from [20].

**Table 4.** Results of comparison of the tabu tenure variants for the random QAP instances (II).
(CPU times per restart are given in seconds. 3 GHz Pentium computer was used in the experiments)

| Instance | *n* | BKV | $\bar{\delta}$ , $C_{1\%}/C_{BKS}$ | | | | | | | | | | | | CPU time |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | FTT-1[d] | | FTT-2[d] | | PTT[d] | | RTT[d] | | DPTT-B[d] | | DPTT-M[d] | | |
| **tai20a** | 20 | 703482 | 0.213 | 10/3 | 0.213 | 10/3 | 0.229 | 10/3 | 0.155 | 10/6 | 0.091 | 10/7 | 0.299 | 10/2 | *0.3* |
| **tai25a** | 25 | 1167256 | 0.316 | 10/2 | 0.118 | 10/7 | 0.231 | 10/4 | 0.195 | 10/5 | 0.159 | 10/6 | 0.246 | 10/4 | *0.7* |
| **tai30a** | 30 | 1818146 | 0.132 | 10/4 | 0.078 | 10/8 | 0.174 | 10/5 | 0.072 | 10/8 | 0.039 | 10/9 | 0.047 | 10/4 | *1.3* |
| **tai35a** | 35 | 2422002 | 0.154 | 10/6 | 0.251 | 10/4 | 0.202 | 10/5 | 0.212 | 10/4 | 0.213 | 10/4 | 0.124 | 10/6 | *2.3* |
| **tai40a** | 40 | 3139370 | 0.435 | 10/0 | 0.558 | 10/0 | 0.477 | 10/0 | 0.273 | 10/0 | 0.395 | 10/0 | 0.471 | 10/0 | *3.8* |
| **tai50a** | 50 | 4941410 | 0.592 | 10/0 | 0.762 | 10/0 | 0.561 | 10/0 | 0.671 | 10/0 | 0.606 | 10/0 | 0.586 | 10/0 | *8.9* |
| **tai60a** | 60 | 7205962 | 0.702 | 10/0 | 0.787 | 10/0 | 0.722 | 10/0 | 0.672 | 10/0 | 0.636 | 10/0 | 0.650 | 10/0 | *18.5* |
| **tai80a** | 80 | 13526696 | 0.543 | 10/0 | 0.741 | 10/0 | 0.526 | 10/0 | 0.537 | 10/0 | 0.573 | 10/0 | 0.506 | 10/0 | *60* |
| **tai100a** | 100 | 21071540 | 0.468 | 10/0 | 0.690 | 10/0 | 0.424 | 10/0 | 0.454 | 10/0 | 0.399 | 10/0 | 0.425 | 10/0 | *147* |

**Table 5.** Results of comparison of the tabu tenure variants for the quasi-random QAP instances (II).
(CPU times per restart are given in seconds. 3 GHz Pentium computer was used in the experiments)

| Instance | *n* | BKV | $\bar{\delta}$ , $C_{1\%}/C_{BKS}$ | | | | | | | | | | | | CPU time |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | FTT-1[d] | | FTT-2[d] | | PTT[d] | | RTT[d] | | DPTT-B[d] | | DPTT-M[d] | | |
| **sko42** | 42 | 15812 | 0.039 | 10/5 | 0.019 | 10/7 | 0.039 | 10/5 | 0.011 | 10/7 | 0.022 | 10/4 | 0.022 | 10/4 | *1.0* |
| **sko49** | 49 | 23386 | 0.061 | 10/3 | 0.071 | 10/1 | 0.085 | 10/0 | 0.079 | 10/0 | 0.044 | 10/3 | 0.044 | 10/3 | *1.7* |
| **sko56** | 56 | 34458 | 0.187 | 10/3 | 0.248 | 10/1 | 0.278 | 10/0 | 0.180 | 10/1 | 0.183 | 10/2 | 0.151 | 10/0 | *2.5* |
| **sko64** | 64 | 48498 | 0.122 | 10/2 | 0.136 | 10/3 | 0.075 | 10/3 | 0.211 | 10/3 | 0.199 | 10/3 | 0.240 | 10/3 | *4.8* |
| **sko72** | 72 | 66256 | 0.139 | 10/1 | 0.069 | 10/2 | 0.129 | 10/0 | 0.241 | 10/0 | 0.177 | 10/0 | 0.125 | 10/0 | *6.6* |
| **sko81** | 81 | 90998 | 0.225 | 10/0 | 0.192 | 10/0 | 0.269 | 10/0 | 0.248 | 10/0 | 0.268 | 10/0 | 0.244 | 10/0 | *10.0* |
| **sko90** | 90 | 115534 | 0.247 | 10/0 | 0.180 | 10/0 | 0.181 | 10/0 | 0.245 | 10/0 | 0.302 | 10/0 | 0.235 | 10/0 | *13.8* |
| **sko100a** | 100 | 152002 | 0.145 | 10/0 | 0.157 | 10/0 | 0.129 | 10/1 | 0.134 | 10/0 | 0.152 | 10/0 | 0.204 | 10/0 | *32* |
| **sko100b** | 100 | 153890 | 0.141 | 10/0 | 0.096 | 10/2 | 0.136 | 10/0 | 0.230 | 10/0 | 0.120 | 10/1 | 0.109 | 10/0 | *33* |
| **sko100c** | 100 | 147862 | 0.200 | 10/0 | 0.156 | 10/3 | 0.195 | 10/0 | 0.042 | 10/0 | 0.156 | 10/0 | 0.115 | 10/0 | *32* |
| **sko100d** | 100 | 149576 | 0.285 | 10/0 | 0.193 | 10/2 | 0.293 | 10/0 | 0.220 | 10/0 | 0.243 | 10/0 | 0.251 | 10/0 | *33* |
| **sko100e** | 100 | 149150 | 0.302 | 10/0 | 0.163 | 10/0 | 0.285 | 10/1 | 0.316 | 10/0 | 0.243 | 10/0 | 0.171 | 10/0 | *33* |
| **sko100f** | 100 | 149036 | 0.260 | 10/0 | 0.230 | 10/0 | 0.274 | 10/0 | 0.339 | 10/0 | 0.231 | 10/0 | 0.249 | 10/0 | *33* |
| **tho30** | 30 | 149936 | 0.069 | 10/7 | 0.023 | 10/9 | 0.214 | 9/8 | 0.255 | 9/5 | 0.149 | 9/8 | 0.121 | 10/5 | *0.4* |
| **tho40** | 40 | 240516 | 0.172 | 10/0 | 0.181 | 10/1 | 0.136 | 10/1 | 0.198 | 10/0 | 0.113 | 10/1 | 0.207 | 10/1 | *0.9* |
| **tho150** | 150 | 8133398 | 0.187 | 10/0 | 0.222 | 10/0 | 0.343 | 10/0 | 0.396 | 10/0 | 0.263 | 10/0 | 0.330 | 10/0 | *174* |
| **wil50** | 50 | 48816 | 0.039 | 10/2 | 0.039 | 10/3 | 0.084 | 10/1 | 0.055 | 10/1 | 0.065 | 10/1 | 0.056 | 10/1 | *1.9* |
| **wil100** | 100 | 273038 | 0.112 | 10/0 | 0.096 | 10/0 | 0.106 | 10/0 | 0.131 | 10/0 | 0.116 | 10/0 | 0.096 | 10/0 | *33* |

**Table 6.** Results of comparison of the tabu tenure variants for the real-life like QAP instances (II).
(CPU times per restart are given in seconds. 3 GHz Pentium computer was used in the experiments)

| Instance | $n$ | BKV | $\bar{\delta}$ , $C_{1\%}/C_{BKS}$ | | | | | | | | | | | | CPU time |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | FTT-1$^d$ | | FTT-2$^d$ | | PTT$^d$ | | RTT$^d$ | | DPTT-B$^d$ | | DPTT-M$^d$ | | |
| **tai20b** | 20 | 122455319 | 0.181 | 10/6 | 0.091 | 10/8 | 0.136 | 10/7 | 0.192 | 10/6 | 0.136 | 10/7 | 0.181 | 10/6 | *0.1* |
| **tai25b** | 25 | 344355646 | 0.060 | 10/7 | 0.138 | 9/6 | 0.014 | 10/8 | 0.022 | 10/7 | 0.573 | 9/7 | 0.604 | 9/5 | *0.2* |
| **tai30b** | 30 | 637117113 | 1.590 | 3/2 | 1.763 | 3/3 | 2.020 | 2/2 | 1.310 | 4/3 | 1.628 | 3/1 | 1.797 | 2/2 | *0.3* |
| **tai35b** | 35 | 283315445 | 0.800 | 8/4 | 2.129 | 4/3 | 0.909 | 7/3 | 1.503 | 6/1 | 0.843 | 8/2 | 1.535 | 6/1 | *0.6* |
| **tai40b** | 40 | 637250948 | 0.840 | 7/6 | 2.636 | 2/2 | 1.278 | 6/5 | 1.972 | 3/2 | 1.972 | 3/3 | 0.846 | 7/7 | *0.9* |
| **tai50b** | 50 | 458821517 | 0.336 | 10/2 | 0.218 | 10/3 | 0.308 | 9/2 | 0.420 | 9/2 | 0.713 | 9/1 | 0.352 | 10/3 | *1.9* |
| **tai60b** | 60 | 608215054 | 0.796 | 5/1 | 1.138 | 3/1 | 0.809 | 5/1 | 1.011 | 4/1 | 0.689 | 6/1 | 0.725 | 6/0 | *4.1* |
| **tai80b** | 80 | 818415043 | 0.930 | 5/0 | 0.949 | 4/0 | 1.027 | 3/0 | 1.058 | 4/0 | 0.976 | 3/0 | 1.027 | 6/0 | *13.0* |
| **tai100b** | 100 | 1185996137 | 0.668 | 7/0 | 0.555 | 8/2 | 0.841 | 7/0 | 1.074 | 6/1 | 0.900 | 7/1 | 0.745 | 7/1 | *33* |
| **tai150b** | 150 | 498896643 | 0.572 | 10/0 | 0.568 | 10/0 | 0.604 | 10/0 | 0.530 | 10/0 | 0.588 | 10/0 | 0.510 | 10/0 | *101* |

**Table 7.** Summary results of comparison of the tabu tenure variants (I)

| Instance type | Deviation from BKV averaged over different types of instances | | | | | |
|---|---|---|---|---|---|---|
| | FTT-1 | FTT-2 | PTT | RTT | DPTT-B | DPTT-M |
| **tai∗a** | 0.371 | 0.462 | 0.376 | **0.348** | 0.368 | 0.400 |
| **sko∗,tho∗,wil∗** | 0.195 | **0.144** | 0.167 | 0.183 | 0.182 | 0.169 |
| **tai∗b** | 0.768 | 0.874 | 0.779 | 0.816 | **0.747** | 0.875 |
| $n < 100$ | 0.433 | 0.461 | 0.426 | 0.433 | **0.402** | 0.459 |
| $n \geq 100$ | 0.296 | 0.318 | **0.280** | 0.304 | 0.327 | 0.314 |
| | 0.393 | 0.419 | 0.383 | 0.394 | **0.380** | 0.416 |

**Table 8.** Summary results of comparison of the tabu tenure variants (II)

| Instance type | Deviation from BKV averaged over different types of instances | | | | | |
|---|---|---|---|---|---|---|
| | FTT-1$^d$ | FTT-2$^d$ | PTT$^d$ | RTT$^d$ | DPTT-B$^d$ | DPTT-M$^d$ |
| **tai∗a** | 0.395 | 0.466 | 0.394 | 0.360 | **0.346** | 0.373 |
| **sko∗,tho∗,wil∗** | 0.163 | **0.137** | 0.181 | 0.196 | 0.169 | 0.165 |
| **tai∗b** | **0.677** | 1.019 | 0.795 | 0.909 | 0.902 | 0.832 |
| $n < 100$ | **0.382** | 0.528 | 0.427 | 0.461 | 0.452 | 0.440 |
| $n \geq 100$ | 0.304 | **0.284** | 0.330 | 0.351 | 0.310 | 0.291 |
| | **0.358** | 0.456 | 0.398 | 0.429 | 0.410 | 0.396 |

**Table 9.** Resulting ranking of the tabu tenure variants

| | FTT-1$^*$ | FTT-2$^*$ | PTT$^*$ | RTT$^*$ | DPTT-B$^*$ | DPTT-M$^*$ |
|---|---|---|---|---|---|---|
| **Deviation from BKV averaged over all instances** | **0.376** | 0.437 | 0.391 | 0.412 | 0.395 | 0.406 |
| **Resulting rank** | **1** | 6 | 2 | 5 | 3 | 4 |

$^*$ the deviations are averaged over the non-delay and delay modifications.

It could also be viewed that, for smaller problems ($n < 100$), it is quite easy to reach the (pseudo-)optimal solutions, while the statistically average quality of solutions is rather bad. On the other hand, it is very difficult to achieve (pseudo-)optimal solutions for larger problems ($n \geq 100$) (if the computation time is limited); however, the average quality of solutions tends to be relatively high.

Our variants of defining the tabu tenure allow to find the near-optimal solutions very quickly (see Tables 1–6). For example, for the large real-life like instance tai150b ($n = 150$), approximately 100 seconds are enough to get the solution that is only 0.5 percent above the pseudo-optimum (see column "FTT-1" of Table 3). The results from Tables 1–6 may be improved considerably by increasing the number of ite-

rations (the run time), or by a more accurate adjusting of the tabu tenures and other control parameters. Thus, we have performed additional extensive experiments on the challenging QAP instance tai100a by using the basic variant of the dynamic tabu tenure with delay. After this long-lasting experimentation – it took several weeks (!) on a 3 GHz computer – we were successful in finding new solution that is slightly better than that announced in [15]. The new objective function value is equal to **21071498**.

## 4. Concluding remarks

In this paper, several variants of defining the tabu tenure for the quadratic assignment problem are discussed. In particular, the fixed tabu tenure, the periodic tabu tenure, the random tenure as well as some variants of the dynamic tabu tenure were investigated. All these variants were examined on various types of the QAP instances taken from the QAP instances library – QAPLIB.

The results from the computational experiments show that is important that the proper values of the tabu tenure (the range of tabu tenure) are maintained. If the tabu tenure is too small, the process returns to the previous solutions time after time. On the other hand, too large tabu tenure may forbid certain promising moves. In any case, the efficiency of the algorithm is lessened significantly. Therefore, a balance between these opposites is needed.

It may also be seen that the deterministic variants of the tabu tenure (in particular, the fixed and periodic tenures) are in most cases superior to the random tabu tenures. The non-fixed (periodic) tabu tenures could possibly be considered to be preferable to the fixed tabu tenures, although the results of experiments with the quasi-random instances indicate that also the fixed tabu tenures hide some potential. So, the further investigation of the fixed tabu tenure as well as other new variants of the tabu tenure (for example, adaptive (self-tuning) tabu tenure) might be interesting. In addition, trying these variants for other combinatorial optimization problems would be worthwhile.

## References


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

**[2]** **N.W. Brixius, K.M. Anstreicher.** The Steinberg wiring problem. *Working Paper, University of Iowa, USA*, 2001.

**[3]** **R.E. Burkard, E.Çela, P.M. Pardalos, L. Pitsoulis.** The quadratic assignment problem. *In D.Z Du, P.M. Pardalos (eds.), Handbook of Combinatorial Optimization, Kluwer, Dordrecht*, 1998, *Vol*.3, 241–337.

**[4]** **R.E. Burkard, S. Karisch, F.Rendl.** QAPLIB – a quadratic assignment problem library. *Journal of Global Optimization*, 1997, *Vol*.10, 391–403. [*See also http://www.seas.upenn.edu/qaplib/*.]

**[5]** **E. Çela.** The Quadratic Assignment Problem: Theory and Algorithms. *Kluwer, Dordrecht*, 1998.

**[6]** **Z. Drezner.** The extended concentric tabu for the quadratic assignment problem. *European Journal of Operational Research*, 2005, *Vol*.160, 416–422.

**[7]** **Z. Drezner, G.A. Marcoulides.** On the range of tabu tenure in solving quadratic assignment problems. *Under review*, 2007.

**[8]** **M. Gendreau.** Recent advances in tabu search. *In C.C Ribeiro, P. Hansen (eds.), Essays and Surveys in Metaheuristics, Kluwer, Boston*, 2001, 369-377.

**[9]** **F. Glover, M. Laguna.** Tabu Search. *Kluwer, Dordrecht*, 1997.

**[10]** **A. Hertz, E. Taillard, D. de Werra.** Tabu search. *In E.H.L. Aarts, J.K. Lenstra (Eds.), Local Search in Combinatorial Optimization, Wiley, Chichester*, 1997, 121–136.

**[11]** **T. Koopmans, M. Beckmann.** Assignment problems and the location of economic activities. *Econometrica*, 1957, *Vol*.25, 53–76.

**[12]** **A. Misevicius.** A tabu search algorithm for the quadratic assignment problem. *Computational Optimization and Applications*, 2005, *Vol*.30, 95–111.

**[13]** **A. Misevičius, J. Blonskis.** Experiments with tabu search for random quadratic assignment problems. *Information Technology and Control*, 2005, *Vol*.34, 237–244.

**[14]** **A. Misevičius, A. Lenkevičius, D. Rubliauskas.** Iterated tabu search: an improvement to standard tabu search. *Information Technology and Control*, 2006, *Vol*.35 (No.3), 187–197.

**[15]** **A. Misevičius, A. Tomkevičius, J. Karbauskas.** Stagnation-protected tabu search variants for unstructured quadratic assignment problems. *Information Technology and Control*, 2006, *Vol*.35 (*No*.4), 363-370.

**[16]** **P.M. Pardalos, F. Rendl, H. Wolkowicz.** The quadratic assignment problem: a survey and recent developments. *In P.M. Pardalos, H. Wolkowicz (eds.), Quadratic Assignment and Related Problems. DIMACS Series an Discrete Mathematics and Theoretical Computer Science, Vol*.16, *AMS, Providence*, 1994, 1-41.

**[17]** **S. Sahni, T. Gonzalez.** P-complete approximation problems. *Journal of ACM*, 1976, *Vol*.23, 555–565.

**[18]** **S. Salhi.** Heuristic search methods. *In G. Marcoulides (ed.), Modern Methods for Business Research, Lawrence Erlbaum, Mahwah*, 1998, 147–175.

**[19]** **E. Taillard.** Robust taboo search for the QAP. *Parallel Computing*, 1991, *Vol*.17, 443–455.

**[20]** **E. Taillard, L.M. Gambardella.** Adaptive memories for the quadratic assignment problem. *Tech. Report* IDSIA-87-97, *Lugano, Switzerland*, 1997.


Received October 2007.