ELSEVIER

European Journal of Operational Research 123 (2000) 256–270

EUROPEAN JOURNAL OF OPERATIONAL RESEARCH

www.elsevier.com/locate/orms

Tabu Search applied to global optimization

Rachid Chelouah [a], Patrick Siarry [b,*]

[a] *Institut Universitaire de Technologie de Cergy-Pontoise, Laboratoire "Modélisation et Optimisation des Systèmes en Electronique", rue d'Eragny, Neuville-sur-Dise, 95031 Cergy-Pontoise, France*

[b] *Université de Paris XII, Faculté des Sciences (LERISS), 61 avenue du Général de Gaulle, 94010 Créteil, France*

Abstract

A new algorithm called Enhanced Continuous Tabu Search (ECTS) is proposed for the global optimization of multiminima functions. It results from an adaptation of combinatorial Tabu Search which aims to follow, as close as possible, Glover's basic approach. In order to cover a wide domain of possible solutions, our algorithm first performs the diversification: it locates the most promising areas, by fitting the size of the neighborhood structure to the objective function and its definition domain. When the most promising areas are located, the algorithm continues the search by intensification within one promising area of the solution space.

The efficiency of ECTS is thoroughly tested by using a set of benchmark multimodal functions, of which global and local minima are known. ECTS is compared to other published versions of continuous Tabu Search and to some alternative algorithms like Simulated Annealing. We point out two main advantages of ECTS: first its principle is rather basic, directly inspired from combinatorial Tabu Search; secondly it shows a good performance for functions having a large number of variables (more than 10).

Keywords: Tabu Search; Global optimization; Continuous variables

1. Introduction

Tabu Search (TS) is a metaheuristic originally developed by Glover [1,2], which has been successfully applied to a variety of combinatorial optimization problems. However, very few works deal with its application to the global minimization of functions depending on continuous variables. Up to now, we are aware of only works [3–6] related to the subject. In this paper, we propose an adaptation of TS to continuous optimization problems, called Enhanced Continuous Tabu Search (ECTS), which is directly inspired from Glover's approach. We produce an efficient algorithm by improving the simplistic approach proposed by Siarry et al. in [6]. To achieve this, we introduce in ECTS diversification and intensification concepts emphasized, in particular, by Glover, and implemented by Battiti et al. [7].

* Corresponding author. Tel.: +33-1-45-17-15-67; fax: +33-1-45-17-14-92.
E-mail address: chelouah@u-cergy.fr (R. Chelouah), siarry@univ-paris12.fr (P. Siarry).

0377-2217/00/$ - see front matter
PII: S 0 3 7 7 - 2 2 1 7 (9 9) 0 0 2 5 5 - 6

First, let us summarize how Glover's idea is adapted to the continuous case in [6]. The algorithm, called CTS, starts from a randomly selected initial solution s. From this current solution s, a set of neighbors, called s', is generated (see details at the beginning of Section 2). To avoid the danger of the appearance of a cycle, the neighbors of the current solution, which belong to a subsequently defined 'tabu list', are systematically eliminated. The objective function to be minimized is evaluated for each generated solution s' and the best neighbor of s becomes the new current solution, even if it is worse than s. After each 'move', s is put into a circular list of tabu solutions: when the tabu list is full, it is updated by removing the first solution entered. Then a new 'iteration' is performed: the previous procedure is repeated by starting from the new current point, until some stopping condition is reached. Usually, the algorithm stops after a given number of iterations without any improvement of the objective function value.

This rudimentary algorithm, directly inspired from the simple combinatorial TS algorithm, is tested through classical multiminima functions of which global and local minima are known [6]. The results obtained are encouraging, but now wholly satisfactory. To improve the approach, it is necessary to carry out diversification (detection of promising areas) allowing the scanning of a wide solution space, and intensification (search inside the most promising area) for a more accurate result. These two operations are described in the current paper.

In order to compare ECTS to other algorithms we have implemented one set of test functions and ECTS, CTS and Enhanced Simulated Annealing (ESA) in the same software. This software is structured in layers and written using the object-oriented language C^{++}. First, we implemented the data structure and then the functional structure. By using these two structures, we developed the various algorithms and the user graphic interface.

The paper is organized as follows. In Section 2, we present the general setting out of the method. In Section 3, ECTS is displayed in detail. The initialization of some parameters and the tuning of the control parameters of ECTS are reported in Section 4. Experimental results are discussed in Section 5 and some words of conclusion make up Section 6.

2. General setting out of the method

First we state the way of discretizing the solution space. We have resumed and adapted the method described in detail in [6]. The neighborhood is defined in [6] by using the concept of 'ball'. A ball $B(s,r)$ is centered on s with radius r; it contains all points s' such that $\|s'-s\| \leqslant r$ (the symbol $\|\cdots\|$ is used to denote the Euclidean norm). To obtain a homogeneous exploration of the space, we consider a set of balls centered on the current solution s, with radius $h_0, h_1, \ldots, h_\eta$. Hence the space is partitioned into concentric 'crowns' $C_i(s, h_{i-1}, h_i)$, such that

$$C_i(s, h_{i-1}, h_i) = \{s' \,|\, h_{i-1} \leqslant \|s'-s\| \leqslant h_i\}.$$

The η neighbors of s are obtained by random selection of one point inside each crown C_i, for i varying from 1 to η. Finally, we select the best neighbor of s among these η neighbors, even if it is worse than s. In ECTS, we replace the balls by hyperrectangles for the partition of the current solution neighborhood (see Fig. 1), and we generate neighbors in the same way. The reason for using a hyperrectangular neighborhood instead of crown 'balls' is the following: it is mathematically much easier to select a point inside a specified hyperrectangular zone than to select a point inside a specified crown ball. Therefore in the first case, we only have to compare the coordinates of the randomly selected point with the bounds that define the hyperrectangular zone at hand.

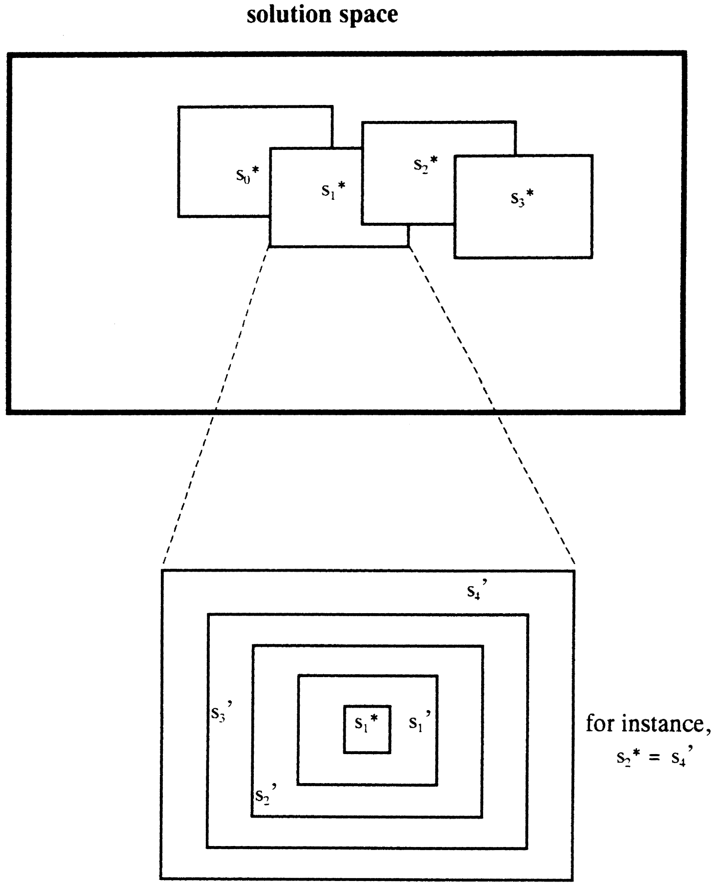

Fig. 1. Partition of current solution neighborhood and diversification strategy.

ECTS consists of the five stages, shown in Fig. 2: setting of parameters, diversification, search for the most promising area, intensification, and output of the best point found. We show below the principle of the three central stages. Similarly to the CTS proposed in [6], ECTS accepts a newly generated neighbor if it is not tabu. ECTS checks that each newly generated neighbor is not inside one of the tabu balls, the centers of which are stored in the tabu list. In order to handle a wide domain of possible solutions, our algorithm performs diversification to locate the most ‘promising areas’. A new promising area is detected whenever an ‘unacceptable’ deterioration (in the sense defined hereafter) of the objective function occurs: if all related deteriorations of the value of the objective function are higher than a given threshold (see details in Section 4), we can then consider that the current solution is at the center of a ball called promising area. The centers of successively obtained promising areas are stored in a list, called the ‘promising list’. To avoid returning to already visited areas, we check that each newly detected ‘promising solution’ is not inside one of the promising balls, the centers of which are saved in the promising list. This condition stimulates the search toward solutions far enough away from the solutions previously obtained.

After a specified number of iterations without any detection of a new promising area, the diversification stops and ECTS determines the ‘most promising area’ of the solution space among the promising areas saved in the promising list. The most promising area is not necessarily the area associated with the best element of the promising list. This particular point is explained in Section 3.

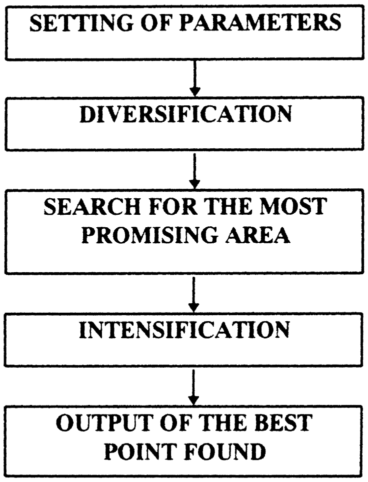

Fig. 2. Simplified structure of ECTS.

ECTS continues by intensifying the search within this most promising area. The center of this area becomes the new current solution, and a continuous tabu search (with iterative generation of neighbors, selection and update) is performed inside that small region of the solution space. The algorithm stops after some given criteria, subsequently defined, are supplied.

3. Detailed presentation of ECTS

In this section, we present the ECTS and introduce several parameters required by the algorithm (‘declaration’ of parameters). But we do not address – or only occasionally – the important question of the value that must be given to those parameters (‘definition’ of parameters). Some of them are initialized at the beginning and other ones are control parameters, which must be carefully tuned before any execution of ECTS. That question is reported in Section 4.

The general flow chart of ECTS is shown in Fig. 3. In this chart, the main stages of ECTS (setting of parameters; diversification; search for the most promising area; intensification; and output of the best point found) are evident. It can be seen that diversification and intensification use the same routines: generation of neighbors, selection of the best neighbor, update of the various lists and adjustment of the parameters. The modular organization of the software in C^{++} takes full advantage of this feature.

3.1. Setting of parameters

Two types of parameters must be set before any execution of ECTS:
- initialization parameters,
- control parameters.

For each of these categories, some parameter values must be chosen by the user and some parameter values must be calculated. These four subsets of parameters are listed in Table 1 and studied in detail in Section 4.

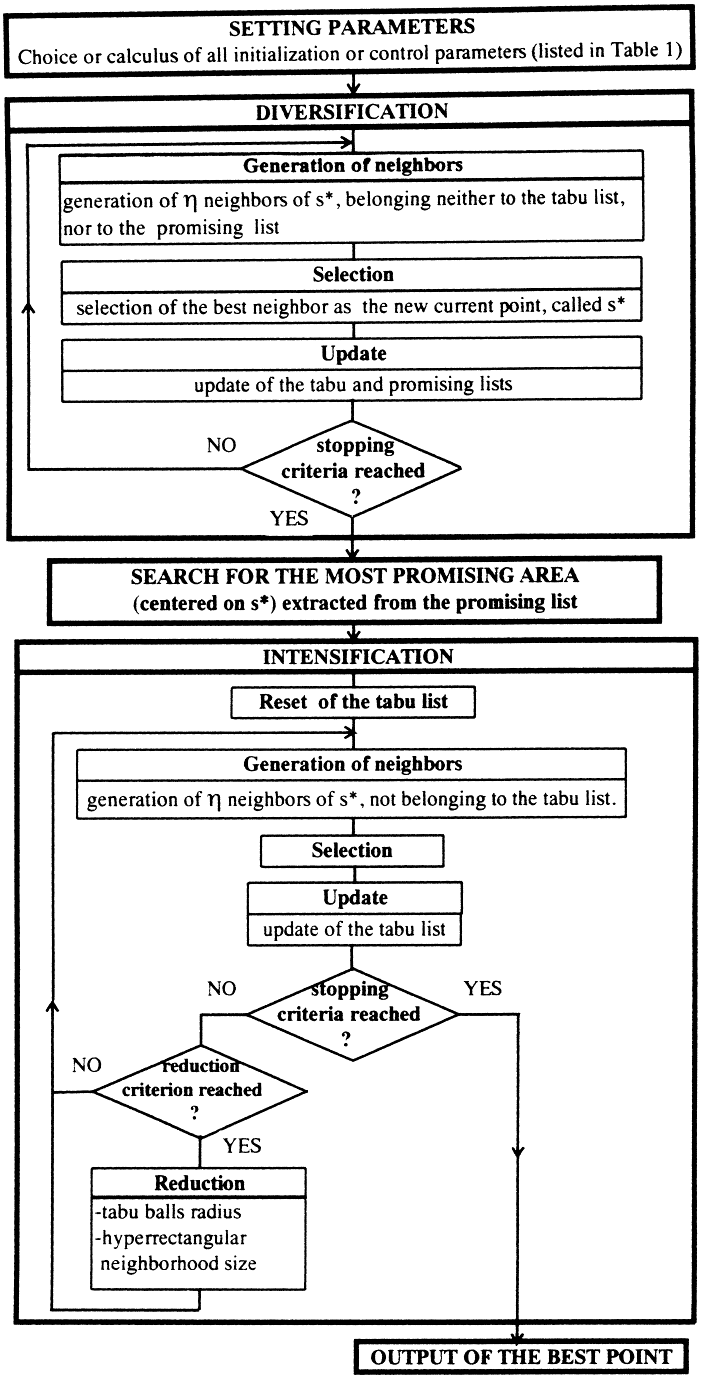

Fig. 3. General flow chart of ECTS.

Table 1
Listing of the ECTS parameters

A. *Initialization parameters chosen by the user*
Search domain of each function variable
Starting point
Content of the tabu list
Content of the promising list
B. *Initialization parameters calculated*
Length δ of the smallest edge of the initial hyperrectangular search domain
Initial threshold for the acceptance of a promising area
Initial best point
Number η of neighbors of the current solution investigated at each iteration
Maximum number of successive iterations without any detection of a promising area
Maximum number of successive iterations without any improvement of the objective function value
Maximum number of successive reductions of the hyperrectangular neighborhood and of the radius of tabu balls without any improvement
Maximum number of iterations
C. *Control parameters chosen by the user*
Length N_t of the tabu list
Length N_p of the promising list
Parameter ρ_t allowing to calculate the initial radius of tabu balls
Parameters ρ_p allowing to calculate the initial radius of promising balls
Parameter ρ_{neigh} allowing to calculate the initial size of the hyperrectangular neighborhood
D. *Control parameters calculated*
Initial radius ε_t of tabu balls
Initial radius ε_p of promising balls
Initial size of the hyperrectangular neighborhood

3.2. Diversification

At this stage, the process starts with the initial solution, used as the current one. ECTS generates a specified number of neighbors: one point is selected inside each hyperrectangular zone around the current solution, as explained in Section 2. Each neighbor is accepted only if it does not belong to the tabu list. The best of these neighbors becomes the new current solution, even if it is worse than the previous one. A new promising solution is detected and generated according to the procedure described above in Section 2. This promising solution defines a new promising area if it does not already belong to a promising ball. If a new promising area is accepted, the worst area of the promising list is replaced by the newly accepted promising area. The use of the promising and tabu lists stimulates the search for solutions far from the starting one and the identified promising areas. The diversification process stops after a given number of successive iterations without any detection of a new promising area (see details in Section 4). Then the algorithm determines the most promising area among those present in the promising list.

3.3. Search for the most promising area

In order to determine the most promising area, we proceed in three steps. First, we calculate the average value of the objective function over all the solutions present in the promising list. Secondly, we eliminate all the solutions for which the function value is higher than this average value. Thirdly, we deal with the thus reduced list in the following way. We halve the radius of the tabu balls and the size of the hyperrectangular neighborhood. For each remaining promising solution, we perform the generation of neighbors and selection of the best. We replace the promising solution by the best neighbor located, yet only if this neighbor is better than that solution. After having scanned the whole promising list, the algorithm removes the least promising solution. This process is reiterated after halving again the above two parameters. It stops when just one promising area remains.

3.4. Intensification

The first step of the intensification stage is the resetting of the tabu list. The remaining promising area allows the definition of a new search domain. The center of this area is taken as the current point, and the tabu search starts again: generation of neighbors not belonging to the tabu list, selection of the best, and insertion of the best solution into the tabu list. This selected neighbor becomes the new current solution, even if it is worse than the previous one. After a predetermined number of successive iterations without any improvement of the objective function value (e.g. quadratic error between two successive solutions less than 10^{-3}), the size of the hyperrectangular neighborhood and the radius of the tabu balls are halved, tabu list is reset, and we restart the procedure from the best point found until now. To stop the algorithm, we use two criteria: a specified number of successive reductions of the two parameters above without any significant improvement of the objective function value and a specified maximum number of iterations. Details about the implementation of these criteria are reported in Section 4.

4. Initialization of some parameters and tuning of the control parameters of ECTS

We describe in this section the initialization of some parameters and the tuning of the control parameters. In other words, we give the 'definition' of all the parameters of ECTS.

The section is divided into two sub-sections. In the first one, we list the parameters which are fixed at the beginning or which are automatically built by using the parameters fixed at the beginning, and we explain how they are valued. In the second one, we enumerate the *control parameters* (which allow to manage the overall behavior of the algorithm), and we explain how they may be tuned.

4.1. Initialization

The parameters fixed at the beginning are the following ones (see Table 1A):
- search domain of each function variable,
- starting point,
- content of the tabu list,
- content of the promising list.

The parameters which are automatically built by using the parameters fixed at the beginning are the following ones (see Table 1B):
- length δ of the smallest edge of the initial hyperrectangular search domain,
- initial threshold for the acceptance of a promising area,
- initial best point,
- number η of neighbors of the current solution investigated at each iteration,
- maximum number of successive iterations without any detection of a promising area,

- maximum number of successive iterations without any improvement of the objective function value,
- maximum number of successive reductions of the hyperrectangular neighborhood and of the radius of tabu balls without any improvement,
- maximum number of iterations.

They are valued in the following way (a brief description of the empirical method used for that purpose is given in Section 4.2:

- the search domain of analytical test functions is set as prescribed in the literature,
- the initial solution s^* is randomly chosen,
- the tabu list is initially empty,
- to complete the promising list, the algorithm randomly draws a point. This point is accepted as the center of an initial promising ball, if it does not belong to an already generated ball. In this way the algorithm generates N_p sample points which are uniformly dispersed in the whole space solution S,
- the initial threshold for the acceptance of a promising area is taken equal to the average of the objective function values over the previous N_p sample points,
- the best point found is taken equal to the best point among the previous N_p sample points,
- the number η of neighbors of the current solution investigated at each iteration is set to twice the number of variables, if this number is equal or smaller than five, otherwise η is set to 10;
- the maximum number of successive iterations without any detection of a new promising area is equal to twice the number of variables,
- the maximum number of successive iterations without any improvement of the objective function value is equal to five times the number of variables,
- the maximum number of successive reductions of the hyperrectangular neighborhood and of the radius of tabu balls without any improvement of the objective function value is set to twice the number of variables,
- the maximum number of iterations is equal to 50 times the number of variables.

4.2. Tuning of the control parameters

There exist two types of control parameters. Some parameters are chosen by the user. Other ones are deduced from the chosen parameters. Both lists are given below.

The control parameters chosen by the user are the following ones (see Table 1C):

- length N_t of the tabu list,
- length N_p of the promising list,
- parameter ρ_t allowing to calculate the initial radius of tabu balls,
- parameter ρ_p allowing to calculate the initial radius of promising balls,
- parameter ρ_{neigh} allowing to calculate the initial size of the hyperrectangular neighborhood.

The control parameters which are calculated are the following ones (see Table 1D):

- initial radius ε_t of tabu balls,
- initial radius ε_p of promising balls,
- initial size of the hyperrectangular neighborhood.

After a series of trials using the analytical test functions listed in Appendix A, we set the value of some of the above-mentioned parameters. We consider that the chosen setting is satisfactory for two reasons:

- this setting represents a compromise over a large set of analytical test functions (20 functions of 2–100 variables; several tests are performed for each function),
- the adopted tunings often are those adopted by other authors, what makes easier the further comparison between competitive algorithms.

Other more specific arguments are given below.

The fixed parameters are the length of the tabu list (set to 7, which is the usual tuning advocated by Glover), the length of the promising list (set to 10, like in [4]) and the parameters ρ_t, ρ_p and ρ_{neigh} (set to 100, 50 and 5, respectively). The expressions of ε_t and ε_p are δ/ρ_t and δ/ρ_p respectively, and the initial size of the hyperrectangular neighborhood of the current solution (the more external hyperrectangle) is obtained by dividing δ by the factor ρ_{neigh}.

5. Experimental results

The efficiency of ECTS was tested using a set of benchmark functions (2–100 variables), which are listed in Appendix A. To obtain a statistically significant comparison of the optimization results, related to the choice of particular starting points, we performed each test 100 times, starting from various randomly selected points in the hyperrectangular search domain given in the usual literature. To this end, the ECTS tests are systematically performed with different initial seeds for the pseudo-random number generator, and hence different starting points.

To specify how long ECTS takes to run, we give an execution time for one function on a personal computer running at 133 MHz: the optimization of *Goldstein–Price* function takes about 3 seconds (average over one hundred executions).

ECTS is tested and compared against other TS implementations and methods. For functions of two to six variables, we compare our results with those previously published. We accept these published results as valid ones, and we do not ourselves program the corresponding algorithms. For functions of more than six variables, there are few available results: consequently ECTS results are only compared to those produced by Siarry et al. in [6,8].

The results of ECTS tests performed on 20 functions are shown in Table 2. To evaluate the efficiency of the algorithm, we retain the following criteria summarizing results from 100 minimizations per function: the rate of successful minimizations, the average number of evaluations of the objective function and the average error. These criteria are defined precisely below.

When at least one of the stopping tests is verified, ECTS stops and provides the coordinates of a located point, and the objective function value '$FOBJ_{ECTS}$' at this point. We compare this result with the known analytical minimum '$FOBJ_{ANAL}$'; and we consider this result to be 'successful' if the following inequality holds:

$$|FOBJ_{ECTS} - FOBJ_{ANAL}| < \varepsilon_{rel} * FOBJ_{ANAL} + \varepsilon_{abs}, \text{ where } \varepsilon_{rel} = 10^{-4} \text{ and } \varepsilon_{abs} = 10^{-6}.$$

The average of the objective function evaluation numbers is evaluated in relation to only the successful minimizations. The average error is defined as the average of FOBJ gaps between the best successful point found and the known global optimum.

The results in Table 2 deserve the following comments. For functions of 2 variables, the average of the objective function evaluation numbers does not exceed 550 with an appreciable accuracy, except for the Easom function, however, the global minimum of the Easom function lies in a very narrow 'hole' of the solution space. Furthermore, for the Shekel functions, ECTS is sometimes trapped into a local minimum. We can avoid this difficulty by increasing the number of iterations without any detection of a promising area or by introducing an aspiration criterion in ECTS. In the case where the problem dimension is between 5 and 9, the rate of successful minimizations is equal to 100%, and the average of objective function evaluation numbers does not exceed 2300. Finally, we test two functions of more than 10 variables: the Zakharov and Rosenbrock functions. The results obtained for the first one are better than those obtained for the second. We explain this difference by the dissymmetry existing between the variables of the Rosenbrock function.

Table 2
Results of ECTS for 20 classical test functions

Test function	Rate of successful minimizations %	Average of objective function evaluation numbers	Average error
RC	100	245	0.05
ES	100	1284	0.01
GP	100	231	2e − 3
SH	100	370	1e − 3
R_2	100	480	0.02
Z_2	100	195	2e − 7
DJ	100	338	3e − 8
$H_{3,4}$	100	548	0.09
$S_{4,5}$	75	825	0.01
$S_{4,7}$	80	910	0.01
$S_{4,10}$	75	898	0.01
R_5	100	2142	0.08
Z_5	100	2254	4e − 6
$H_{6,4}$	100	1520	0.05
R_{10}	85	15 720	0.02
Z_{10}	100	4630	2e − 7
R_{50}	75	63 210	0.02
Z_{50}	100	63 970	2e − 7
R_{100}	75	162 532	0.05
Z_{100}	100	152 030	1e − 3

The performance of ECTS is then compared to four other published versions of continuous TS and alternative algorithms, in particular Simulated Annealing. These various methods are listed in Table 3. The difference between $CRTS_{min}$ and $CRTS_{ave}$ is as follows: in $CRTS_{min}$, the algorithm uses the minimum FOBJ value among the neighbors under consideration, whereas, in $CRTS_{ave}$, the algorithm uses the average FOBJ value.

The experimental results obtained for 12 test functions of less than 10 variables, using the 7 different methods, are given in Table 4: for each function, we give the average number of function evaluations for 100 runs. It can be seen that some results are not available for some methods. Indeed, these methods [4,5,9] have not been tested at all on some of the benchmark instances: TS [4] is tested on six test functions, Continuous Reactive Tabu Search [5] and INTEROPT [9] are tested on the same set of seven test functions, while our algorithm ECTS is tested on 20 test functions. The numbers in parentheses are the

Table 3
Listing of various methods used in the comparison

Method	Reference
Enhanced Continuous Tabu Search (ECTS)	This work
Continuous Tabu Search (CTS)	Siarry and Berthiau [6]
Continuous Reactive Tabu Search (CRTS min)	Battiti and Tecchiolli [5]
Continuous Reactive Tabu Search (CRTS ave)	Battiti and Tecchiolli [5]
Taboo Search (TS)	Cvijovic and Klinowski [4]
Enhanced Simulated Annealing (ESA)	Siarry et al. [8]
INTEROPT	Bilbro and Snyder [9]

Table 4
Average number of objective function evaluations used by seven methods to optimize 12 functions of less than 10 variables[a]

Function	Method						
	ECTS	CTS	CRTS min	CRTS ave	TS	ESA	INTEROPT
RC	245	668	41	38	492	–	4172
GP	231	1636	171	248	486	783	6375
SH	370	1123	–	–	727	–	–
R_2	480	1616	–	–	–	796	–
Z_2	195	689	–	–	–	15 820	–
$H_{3,4}$	548	628	609	513	508	698	1113
$S_{4,5}$	825 (0.75)	4889 (0.5)	664	812	–	1137 (0.54)	3700 (0.4)
$S_{4,7}$	910 (0.8)	5110 (0.6)	871	960	–	1223 (0.54)	2426 (0.6)
$S_{4,10}$	898 (0.75)	5232 (0.56)	693	921	–	1189 (0.5)	3463 (0.5)
R_5	2142	52 733	–	–	–	5364	–
Z_5	2254	27 659	–	–	–	69 799	–
$H_{6,4}$	1520	2250	1245	750	2845	2638	17 262

[a] The numbers in parentheses are the ratios of runs for which the algorithm found the global minimum rather than being trapped into a local minimum.

ratios of runs for which the algorithm found the global minimum rather than being trapped into a local minimum.

Thus the single criterion retained for the comparison between the various algorithms is the average number of function evaluations (evaluated for 1000 runs in standard time units in [5,9] and for 100 runs for all the remaining methods). Tables 4 and 5 represent complete data about the comparison of the algorithmic load.

Other comparisons concerning the quality of the produced solutions are not possible, because this question is not addressed at all by the other authors, except in [5], where it is only said that 'the statistical error on the CRTS averages is about 3%'.

The quality of the solutions produced by ECTS is displayed in Table 2.

The results from ECTS are far better than the results from CTS, our previous version of continuous TS. ECTS results are satisfactory for all of the functions. For instance, the average computing cost of ECTS is equal to 231 function evaluations for the Goldstein–Price function, GP, and to 548 for the Hartmann function, $H_{3,4}$. These results are close to the best available results, obtained by Battiti et al. [5]: for GP, 171 evaluations with CRTS_{min} and 248 evaluations with CRTS_{ave}; and for $H_{3,4}$, 609 evaluations with CRTS_{min} and 513 evaluations with CRTS_{ave}.

Table 5
Average number of objective function evaluations used by three methods to optimize six functions of 10–100 variables[a]

Function	Method		
	ECTS	CTS	ESA
R_{10}	15 720 (0.85)	263 299 (0.7)	12 403
Z_{10}	4630	295 820 (0.5)	15 820
R_{50}	63 210 (0.75)	>10E+6	78 224
Z_{50}	63 970	>10E+6	195 726
R_{100}	162 532 (0.75)	>10E+6	188 227
Z_{100}	152 030	>10E+6	789 718

[a] The numbers in parentheses are the ratios of runs for which the algorithm found the global minimum rather than being trapped into a local minimum.

All the methods, except CRTS, occasionally failed to detect the global minimum of the Shekel functions; although the rate of successful minimizations is best for ECTS.

Table 5 shows the comparison of ECTS to CTS [6] and ESA [8] for functions of 10–100 variables (no data are available for other published methods for this type of function), and it appears that our algorithm provides the best results.

6. Conclusion

In this paper, we show that TS can be efficiently applied to the optimization of continuous multiminima functions. Our main contribution is the adaptation to the continuous case of two concepts widely used in combinatorial TS, namely diversification and intensification. We propose diversification for the detection of promising areas, and intensification as a search within the most promising area.

The results are satisfactory, and for functions having less than 10 variables, we obtain similar or better results than the ones provided by other methods or other versions of continuous TS. We avoid performing CPU time costly partitions of the solution space and using approaches too sophisticated for the neighborhood structure. So, as we can see with results discussed in Section 5, ECTS can be applied to functions having a large number of variables without the prohibitive increase of CPU time. Furthermore, ECTS is much simpler than other versions of continuous TS because it is naturally derived from combinatorial TS.

With regard to the future, we believe that the management of the promising list and the strategy for detection of promising areas could be further improved. CPU time could be drastically reduced by using a parallel version of ECTS. Promising areas, e.g., could be detected by several processors working in parallel.

Appendix A. List of test functions

Branin RCOS (RC) (2 variables):

- $\mathrm{RC}(x_1,x_2) = (x_2 - (5/4\pi^2))x_1^2 + (5/\pi)x_1 - 6)^2 + 10(1 - (1/(8.\pi)))\cos(x_1) + 10$;
- search domain : $-5 < x_1 < 10, 0 < x_2 < 15$;
- no local minimum;
- three global minima: $(x_1,x_2)^* = (-\pi, 12.275), (\pi, 2.275), (9.42478, 2.475)$;
- $\mathrm{RC}((x_1,x_2)^*) = 0.397887$.

Easom (ES) (2 variables):

- $\mathrm{ES}(x_1,x_2) = -\cos(x_1)\cos(x_2)\exp(-((x_1-\pi)^2 + (x_2-\pi)^2))$;
- search domain: $-100 < x_j < 100,\ j = 1,2$;
- several local minima (exact number unspecified in the usual literature);
- one global minimum: $(x_1,x_2)^* = (\pi,\pi)$; $\mathrm{ES}((x_1,x_2)^*) = -1$.

Goldstein and Price (GP) (2 variables):

- $$\begin{aligned}\mathrm{GP}(x_1,x_2) = {} & [1 + (x_1+x_2+1)^2 * (19 - 14x_1 + 13x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \\ & * [30 + (2x_1 - 3x_2)^2 * (18 - 32x_1 + 12x_1^2 - 48x_2 - 36x_1x_2 + 27x_2^2)];\end{aligned}$$
- search domain : $-2 < x_j < 2,\ j = 1,2$;
- four local minima; one global minimum: $(x_1,x_2)^* = (-1,0)$; $\mathrm{GP}((x_1,x_2)^*) = 3$.

Shubert (SH) (2 variables):

$$\mathrm{SH}(x_1,x_2) = \left\{ \sum_{j=1}^{5} j\cos[(j+1)x_1 + j] \right\} * \left\{ \sum_{j=1}^{5} j\cos[(j+1)x_2 + j] \right\};$$

- search domain: $-10 < x_j < 10, \ j = 1,2$;
- 760 local minima;
- 18 global minima: SH $((x_1,\ x_2)^*) = -186.7309$.

De Joung (DJ) (3 variables):
- $\mathrm{ES}(x_1,x_2,x_3) = x_1^2 + x_2^2 + x_3^2$; search domain: $-5.12 < x_j < 5.12, \ j = 1,3$;
- one single minimum (local and global) : $(x_1,x_2,x_3)^* = (0,0,0)$; $\mathrm{ES}((x_1,x_2,x_3)^*) = 0$.

Hartmann ($H_{3,4}$) (3 variables):

$$H_{3,4}(\boldsymbol{x}) = -\sum_{i=1}^{4} c_i \exp\left[-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2 \right];$$

- search domain: $0 < x_j < 1, \ j = 1,3$;
- four local minima: $\boldsymbol{p}_i = (p_{i1},p_{i2},p_{i3}) = i$th local minimum approximation; $f((\boldsymbol{p}_i)) \cong -c_i$;
- one global minimum: $\boldsymbol{x}^* = (0.11, 0.555, 0.855)$; $H_{3,4}(\boldsymbol{x}^*) = -3.86278$.

i	a_{ij}			c_i	p_{ij}		
1	3.0	10.0	30.0	1.0	0.3689	0.1170	0.2673
2	0.1	10.0	35.0	1.2	0.4699	0.4387	0.7470
3	3.0	10.0	30.0	3.0	0.1091	0.8732	0.5547
4	0.1	10.0	35.0	3.2	0.0381	0.5743	0.8828

Shekel ($\boldsymbol{S_{4,n}}$) (4 variables):

$$S_{4,n}(\boldsymbol{x}) = -\sum_{i=1}^{n} \left[(\boldsymbol{x} - \boldsymbol{a}_i)^{\mathrm{T}}(x - a_i) + c_i \right]^{-1};$$

- $\boldsymbol{x} = (\boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{x_3}, \boldsymbol{x_4})^{\mathrm{T}}$;
- $\boldsymbol{a}_i = (a_i^1, a_i^2, a_i^3, a_i^4)^{\mathrm{T}}$;
- three functions $S_{4,n}$ were considered: $S_{4,5}, S_{4,7}$ and $S_{4,10}$;
- search domain: $0 < x_j < 10, \ j = 1, \ldots, 4$;
- n local minima ($n = 5, 7$ or 10): $\boldsymbol{a}_i^{\mathrm{T}} = i$th local minimum approximation: $S_{4,n}((\boldsymbol{a}_i^{\mathrm{T}})) \cong -1/c_i$;
- one global minimum: $\boldsymbol{x}^* = (4,4,4,4)$; $S_{4,n}(\boldsymbol{x}^*) = -10.40294$.

i	$\boldsymbol{a}_i^{\mathrm{T}}$				c_i
1	4.0	4.0	4.0	4.0	0.1
2	1.0	1.0	1.0	1.0	0.2
3	8.0	8.0	8.0	8.0	0.2
4	6.0	6.0	6.0	6.0	0.4
5	3.0	7.0	3.0	7.0	0.4

Table (Continued)

i	$\boldsymbol{a}_i^{\mathrm{T}}$				c_i
6	2.0	9.0	2.0	9.0	0.6
7	5.0	5.0	3.0	3.0	0.3
8	8.0	1.0	8.0	1.0	0.7
9	6.0	2.0	6.0	2.0	0.5
10	7.0	3.6	7.0	3.6	0.5

Hartmann ($\boldsymbol{H_{6,4}}$) (*6 variables*):

$$\mathrm{H}_{6,4}(\boldsymbol{x}) = -\sum_{i=1}^{4} c_i \exp\left[-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2\right];$$

- search domain: $0 < x_j < 1,\ j = 1, 6$;
- four local minima: $\boldsymbol{p_i} = (p_{i1}, \ldots, p_{i6})$ = ith local minimum approximation; $f((\boldsymbol{p_i})) \cong -c_i$.

i	a_{ij}						c_i	p_{ij}					
1	10.0	3.00	17.0	3.50	1.70	8.00	1.0	0.1312	0.1696	0.5569	0.0124	0.8283	0.5886
2	0.05	10.0	17.0	0.10	8.00	14.0	1.2	0.2329	0.4135	0.8307	0.3736	0.1004	0.9991
3	3.00	3.50	1.70	10.0	17.0	8.00	3.0	0.2348	0.1451	0.3522	0.2883	0.3047	0.6650
4	17.0	8.00	0.05	10.0	0.10	14.0	3.2	0.4047	0.8828	0.8732	0.5743	0.1091	0.0381

Rosenbrock ($\boldsymbol{R_n}$) (*n variables*):

$$R_n(\boldsymbol{x}) = \sum_{j=1}^{n-1}\left[100(x_j^2 - x_{j+1})^2 + (x_j - 1)^2\right];$$

- five functions were considered: R_2, R_5, R_{10}, R_{50} and R_{100};
- search domain: $-5 < x_j < 10,\ j = 1, \ldots, n$;
- several local minima (exact number unspecified in the usual literature);
- one global minimum: $\boldsymbol{x}^* = (1, \ldots, 1);\ R_n(\boldsymbol{x}^*) = 0$.

Zakharov ($\boldsymbol{Z_n}$) (*n variables*):

$$Z_n(\boldsymbol{x}) = \left(\sum_{j=1}^{n} x_j^2\right) + \left(\sum_{j=1}^{n} 0.5jx_j\right)^2 + \left(\sum_{j=1}^{n} 0.5jx_j\right)^2;$$

- five functions were considered: Z_2, Z_5, Z_{10}, Z_{50} and Z_{100};
- search domain: $-5 < x_j < 10,\ j = 1, \ldots, n$;
- several local minima (exact number unspecified in the usual literature);
- one global minimum: $\boldsymbol{x}^* = (0, \ldots, 0);\ Z_n(\boldsymbol{x}^*) = 0$.

References

[1] F. Glover, Tabu search: Part I, ORSA Journal on Computing 1 (3) (1989) 190–206.

[2] F. Glover, Tabu search: Part II, ORSA Journal on Computing 2 (1) (1990) 4–32.

[3] N. Hu, Tabu search method with random moves for globally optimal design, International Journal for Numerical Methods in Engineering 35 (1992) 1055–1070.

[4] D. Cvijovic, J. Klinowski, Taboo search: An approach to the multiple minima problem, Science 667 (1995) 664–666.
[5] R. Battiti, G. Tecchiolli, The continuous reactive tabu search: Blending combinatorial optimization and stochastic search for global optimization, Annals of Operations Research 63 (1996) 53–188.
[6] P. Siarry, G. Berthiau, Fitting of tabu search to optimize functions of continuous variables, International Journal for Numerical Methods in Engineering 40 (1997) 2449–2457.
[7] R. Battiti, G. Tecchiolli, The reactive tabu search, ORSA Journal on Computing 6 (2) (1994) 126–140.
[8] P. Siarry, G. Berthiau, F. Durbin, J. Haussy, Enhanced simulated annealing for globally minimizing functions of many continuous variables, ACM Transactions on Mathematical Software 23 (2) (1997) 209–228.
[9] G.L. Bilbro, W.E. Snyder, Optimization of functions with many minima, IEEE Transactions on Systems, Man, and Cybernetics 21 (4) (1991) 840–849.