Branch and Bound Hybrid Algorithm to Determine the Exact or Approximate Solution of the 0/1 Knapsack Problem with One Parameter

José Crispín Zavala-Díaz[1], Marco Antonio Cruz-Chavez[2], Martín H. Cruz-Rosales[3], Juan Frausto-Solís[4]

[1]FCAeI, [2]CIICAp, [3]FC, Autonomous University of Morelos State

[4]Department of Computer Science, ITESM, Campus Cuernavaca

{crispin_zavala, mcruz,mcr}@uaem.mx, juan.frausto@itesm.mx

Abstract

In this paper, the Zavala-Cruz Algorithm is presented to solve the knapsack problem with one parameter. The foundation of the algorithm is shown; it is based largely on the algorithm of Horowitz-Sahni, with the lower and upper bounds taken from Dantzig, and the artifices of simulated annealing used to in order to escape of the local optimums. The Zavala-Cruz Algorithm defines the search space and the rules to branch and prune, with which avoids the backtracking and this accelerates quicker the convergence to an exact or approximate solution. The Zavala-Cruz Algorithm determines the exact solution for all the uncorrelated instances and for some of the weakly correlated instances. It also determines the approximate solution for the strongly correlated instances. These solutions were obtained along the whole parameter, including the transition phase, where the most complex instances are found, and where in some cases the instances are not computable. For each instance and for each value of the parameter, a multiple of n^2 *was used in order to branch and prune, obtaining convergence with* kn^2 *iterations in all cases.*

1. Introduction

Several investigators since 1950 have developed methods and algorithms to determine the exact or approximate solution to the 0/1 knapsack problem. When the size and/or the complexity of new instances increases, in search of a solution, the scientific community either proposes modifications to the existing algorithms, or creates new algorithms. This paper presents a hybrid algorithm to solve discrete optimization problems with a parameter. The structure of the new algorithm is based on the Horowitz-Sahni [1] algorithm, the Lower and Upper Bounds on the work of Dantzig [1], and the artifices to escape of the local optimums from the simulated annealing heuristic [2]. In this algorithm, called the Zavala-Cruz (ZC) algorithm, the procedures are defined to reduce the search space and the criterions in order to branch and bound.

In order to show the ZC algorithm, 0/1 knapsack problems with one parameter are solved. Their formulation is the following:

$$\max\ z = \sum_{i=1}^{n} p_i x_i \qquad \textbf{(1)}$$

$$\text{Subject to: } \sum_{i=1}^{n} w_i x_i \le \lambda C, \qquad x_i \in \{0,1\} \qquad \textbf{(2)}$$

$$\text{Where: } C = \sum_{i=1}^{n} w_i, \quad 0 \le \lambda \le 1, p_i, \quad w_i \in Z^+ \qquad \textbf{(3)}$$

In the formulation of the problem, the optimum z is obtained for each value of λ multiplied by C, where $0 \le \lambda \le 1$, with p_i as the profit, w_i as the weight of the element i, and C as the sum of the weight of all the elements. The i element is part of the solution when $x_i = 1$ and it is not part of the solution when $x_i = 0$.

In (2), for each value of λ, an instance of the 0/1 knapsack problem is solved. Each instance is a problem that is considered NP-hard [3].

To understand the behavior of the algorithm along the parameter λ it is not necessary to solve the whole continuous interval [0, 1], it is only necessary to identify the phases of transition [4, 5]. For the 0/1 knapsack problem, the phase of transition is considered to be when $\lambda = 0.5$, and for that reason, the computational experimentation was done with restriction values where the value of $\lambda = 0.5$ [1, 6]. In this paper, it is demonstrated that the position of the phase of transition depends on the type of problem that is solved.

The ZC algorithm uses two variables in order to control the iterative process. The first, and optimal option, is to determine an exact solution. In the case that the exact solution is not determined within the limit of considered iterations, the value of the optimum solution is increased so that the number of branches is pruned. Thus, an approximate solution is converged upon in a reasonable number of iterations, kn^2, where n is the size of the problem.

In the computational experimentation of the knapsack problem, the types of problems used are grouped according to their complexity as a function of the coefficients w_i and p_i. Based on these coefficients, problems are classified into three groups: uncorrelated, weakly correlated, and strongly correlated. The ZC algorithm determined the exact solution for the uncorrelated problems and determined the approximate solution for the strongly correlated problems. For the weakly correlated problems, the exact solution for some instances was determined, and for the other instances the approximate solution was determined.

The second section of the paper defines the solution process and presents the Zavala-Cruz Algorithm. The third section describes how the search space is generated. The fourth section presents and discusses the obtained results. Finally, the fifth section states the conclusions.

2. Solution Process

It is evident that no process is necessary when:

- $\lambda = 0$, the solution of the problem is $X(0, 0, ..., 0)$ and the maximum $z = \sum_{i=1}^{n} p_i x_i = 0$
- $\lambda = 1$, the solution of the problem is $X(1, 1, ..., 1)$ and the maximum $z = \sum_{i=1}^{n} p_i$

Since the values at the two extremes of the interval are well-known, it is only necessary to work with the interval λ_0, where $0<\lambda_0<1$, and calculate $c_0 = \lambda_0 C$. The solution is obtained with the coefficients $\frac{p_1}{w_1}, \frac{p_2}{w_2}, ..., \frac{p_n}{w_n}$ as Dantzig [7] proposes, ordered from greatest to smallest amount:

$\frac{p_{i1}}{w_{i1}} \geq \frac{p_{i2}}{w_{i2}} \geq ... \geq \frac{p_{in}}{w_{in}}$, and the critical variable x_s is determined:

$$s = \min\left\{ l_0 : \sum_{k=1}^{l_0} w_{ik} > c_0 \right\} \tag{4}$$

The Upper Bound (UB*)* is defined in the following form:

$$UB = \left\lfloor \sum_{k=1}^{l_0-1} p_{ik} + \left(c_0 - \sum_{k=1}^{l_0-1} w_{ik} \right) \frac{p_{i_{l_0}}}{w_{i_{l_0}}} \right\rfloor \tag{5}$$

The Lower Bound (LB) is defined for the first term of the equation (5) as:

$$LB = \sum_{k=1}^{l_0-1} p_{ik} \tag{6}$$

As in the Horowitz-Sahni algorithm (H-S) [1], the ZC Algorithm begins with a partial solution, where the UB and LB limits are obtained. Later on, as in the H-S algorithm, a search is carried out by means of reduction of the elements in the initial set. In each one of these new search spaces the new UB and LB are determined.

The first LB is the first integer optimum solution (z). Two offspring are generated either by fixing an element in the solution or by removing it, as the following explanation describes. The $UB_{offspring}$ of the two generated offspring are used in order to determine if the tree should branch or prune. To branch in the direction of the new offspring, it must be true that $UB_{offspring} > z$. This indicates the possibility that a better $LB_{offspring}$ than the current one exists, and consequently a better optimum solution could be found. If it does not branch in the direction of the offspring, it is pruning, which means that a better $LB_{offspring}$ does not exists that is greater that LB. In this way, the search spaces that form a binary tree are generated, as described in section 2.2.

To obtain a binary tree of smaller size, apart from the rule of branching and pruning, the artifice can be applied from the simulated annealing (SA) metaheuristic [8], by increasing the temperature in order to leave a quasi-stationary state. This artifice is simply the application of a restart in SA [8, 9], that is based on an increase in the parameter of control (cooling temperature in SA) which breaks the quasi-stationary state, thereby allowing escape from the local optimum [8]. In the ZC Algorithm (see section 2.1), the increase of the control parameter is carried out by the coefficient ε.

The control parameter in SA is the variable that is used to define the search space where the optimum solution may be [8]. The control parameter is involved in the Boltzmann function, which is used in SA to decide whether to accept or reject a configuration (a solution to the problem) where the optimal solution could be part of the neighborhood of the configuration. If the control parameter has a very high value, upon evaluating this value in the Boltzmann function, the probability of acceptance of the new configuration is very high. The opposite happens when the value of the control parameter is very small. In the ZC Algorithm, the best solution found (z) is the control parameter either to accept (branch) or to reject (prune) configurations. Therefore, it is the magnitude of (z) that increases by means of the parameter ε, that depends on the number of given iterations by *h* (see section 2.1). This is expressed in equation 7.

$$UB > z(1+\varepsilon) \tag{7}$$

2.1. Zavala-Cruz algorithm

If $\varepsilon = 0$, the exact solution will be obtained. If $\varepsilon > 0$ the approximate solution will be found, and the limits within which the optimal solution is found will be known.

Zavala-Cruz´s algorithm

```
Input $n, w_i, p_i$,
Output $z, UB, \varepsilon$.

 $z = 0, \varepsilon = 0, iteration = 1, h= 0, \Delta\varepsilon \neq 0, \Delta iter \neq 0,$

$limit\_iteration = \Delta iter$
$convergence = false$

While not convergence {
  $LB = \sum_{k=1}^{l_0-1} p_{ik}$

  $UB = \left\lfloor \sum_{k=1}^{l_0-1} p_{ik} + \left(c_0 - \sum_{k=1}^{l_0-1} w_{ik}\right) \frac{p_{i_{l_0}}}{w_{i l_0}} \right\rfloor$

  if limit_iteration > iteration {
     $\varepsilon = h * \Delta\varepsilon$
  }
  else {
       $h = h + 1$
       $limit\_iteration = limit\_iteration + \Delta iter$
  }
  if $UB > z*(1+ \varepsilon)$ {
     if $LB > z$ { $z = LB$ }
      Branch=
      (Generate two offspring$(w_i, p_i)_{LEFF}$, $(w_i, p_i)_{RIGHT}$)
      Change $(w_i, p_i)$ =
     ($(w_i, p_i)_{LEFF}$ or $(w_i, p_i)_{RIGHT}$)     // randomly selected
     Stock up in heap =
   ($(w_i, p_i)_{LEFF}$ or $(w_i, p_i)_{RIGHT}$)    //the selected number
  }
  else {
     If heap = empty {
        Convergence = true
     }
     else {              //obtained of the heap with LIFO
          Change $(w_i, p_i)$ = ($(w_i, p_i)_{LEFF}$ or $(w_i, p_i)_{RIGHT}$)
     }
  }
  Iteration = iteration + 1
}
```

2.2. Search Space Generation

The search space of the initial problem is considered for the two subsets of the power set that is formed with the elements of the initial problem. If the elements of the initial problem are in the set $I = \{1, 2, ..., m\}$, their power set will have the number of subsets $|P(I)| = 2^m$. This power set is represented for a cube formed by the empty set $\varnothing$ and the set *I*, this cube is indicated as $C(m) = [\varnothing, I]$.

According to the ZC algorithm two offspring are generated in the following level, these offspring are two disjunctive cubes: $C_1(m-1)$ and $C_2(m-1)$. Each cube has the dimension $(m-1)$. These cubes are made in the following way. The first element is fixed to the left offspring, and the first element is removed from the right. The search space of the left offspring is represented by the cube $C_1(m-1)$ with an interval $[\{1\}, I]$. The search space of the right offspring is represented by the cube $C_2(m-1)$ with an interval $[\varnothing, I\backslash\{1\}]$. These two cubes are disjunctive subsets, $C_1(m-1)\cap C_2(m-1) = \varnothing$, because all the elements of the cube $C_1(m-1)$ have the subset $\{1\}$ and none of the elements of the cube $C_2(m-1)$ the subset $\{1\}$ [10]. This means that in the left offspring, the first element will be part of the solution, while in the right offspring it will not. Later, the lower levels of the tree are formed $(m-2)$, and the same operations are performed in both cubes, C_1 $(m-1)$ and C_2 $(m-1)$. Each cube divides itself into two cubes of smaller dimensions by fixing or removing the first free element. At the level *(m-2)* there are four cubes. If this process is continued it can be seen that at the level *m-3* there are eight cubes, two cubes for each one of the level *m-2* cubes. In general, for the level *(m-k)*, $0 \leq k \leq m$, 2^k cubes are formed, where each two cubes are disjunctives sets and the union of all the cubes at a given level forms the initial cube *C(m)*. All the cubes of the level *(m-k)* are a part of the cube *C(m)*.

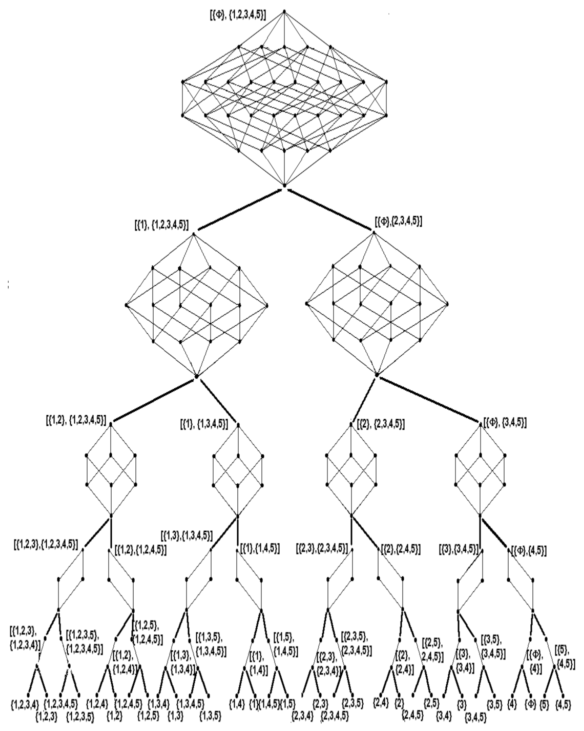

Fig. 1. Binary tree of the search space for $m = 5$

For the last level, the number of cubes of zero dimension is equal to 2^m. This is then equal to the number of vertexes of the initial cube *C(m)*. The set is the partition because each intersection is equal to zero, and the union of them is the initial set *C(m)*. Figure 1, the complete tree that is generated with the search space of the ZC algorithm for $m= 5$ is shown, without the use of the rules of branch and prune. It is clear that the trees that are generated with the ZC algorithm are not complete because of the rules and the artifice of simulated annealing.

3. Experimental results

The ZC algorithm was carried out using ANSI C, and the tests were carried out using an HP Proliant with a 3.2 Ghz Xeon Intel processor. The operating system was Fedora 5.0.

The computational experimentation of the ZC algorithm was carried out while considering the classification of problems proposed by Martello, et al. [6]. They classify the instances as a function of the relationship that exists between the lineal solution and the lineal integer solution. If these two solutions are very close, the problem is classified as uncorrelated or weakly correlated, and all of their combinations. If the two solutions are not close, the problem is classified as strongly correlated and all its combinations. In either case, the values of the coefficients are determined randomly with the following restrictions:

- Uncorrelated: $w_i \in [1, R]$ and $p_i \in [1, R]$
- Weakly correlated: $w_i \in [1, R]$ and $p_i \in \left[w_i - \frac{1}{10}R, w_i + \frac{1}{10}R \right]$, such that $p_i \geq 1$
- Strongly correlated: $w_i \in [1, R]$ and $p_i = w_i + \frac{1}{10}R$
- Inverse strongly correlated: $w_i = p_i + \frac{1}{10}R$ and $p_i \in [1, R]$
- Almost strongly correlated: $w_i \in [1, R]$ and $p_i \in \left[w_i + \frac{1}{10}R - \frac{1}{500}R, w_i + \frac{1}{10}R + \frac{1}{500}R \right]$
- Subset sum: $w_i \in [1, R]$ and $p_i = w_i$
- Uncorrelated with similar weights: $w_i \in [100000, 100100]$ and $p_i \in [1, 1{,}000]$

For each one of the classified problems, a lineal solution, lineal integer solution, and the elements of the optimum solution for each λ_0 selected were obtained, in such a way that the whole interval $0 < \lambda < 1$ was covered. The size of the instances used was $m = 500$ and $R = 1000$, where $R \in \{1{,}000, 10{,}000\}$.

In the Zavala-Cruz algorithm, the control of the iterative process is by means of the *limit_iteration* variable and the coefficient ε. The values of these two variables depends on the following: if all the search spaces of the binary tree are generated, all the vertexes in each one of their levels will be obtained, and this total will be equal to the sum of the vertexes of each level: $1 + 2 + 4 + \ldots + 2^m$. The vertexes can be described as $2^{m+1} - 1$. If the search is carried out in each one of the vertexes, it will have a complexity of $O(2^{m+1})$, which makes the problem incomputable and unable to be solved. It has been demonstrated theoretically that the complexity of solving the knapsack problem by means of the generation of instances with random numbers is between $O(n^2)$ and $O(kn^5)$ [11]. Consequently, the number of iterations necessary in order to converge is between those two limits. Therefore, the inferior limit is selected as the beginning measurement from which to increase the *limit_iteration* until convergence is arrived upon.

Considering the previous description and the previous computational experiments of the investigators [12], the inferior limit from which to begin the calculations is kn^2, where k is an integer. In this case, k will be equal to five because with this multiple, the exact solutions of the uncorrelated problems are determined. When the number of iterations arrives at the first limit, it adds kn^2 iterations to increase the limit in order to converge. It also increases ε to *0.0005*. In the Figures 2 through 8, the number of iterations necessary to converge in each one of the mentioned cases is shown.

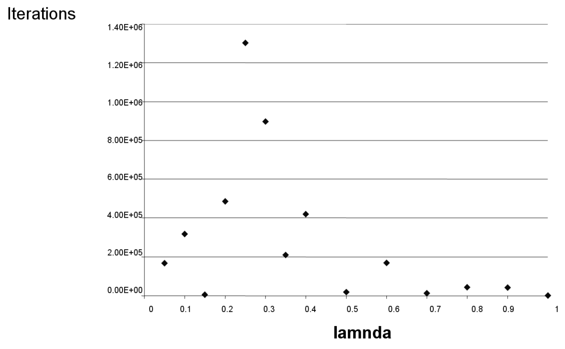

Fig. 2. Instance of the uncorrelated problem.

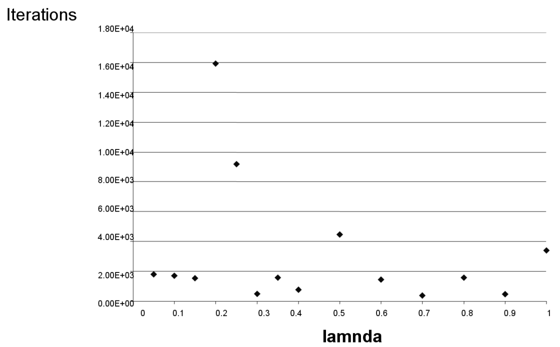

Fig. 3. Instance of the Subset sum problem.

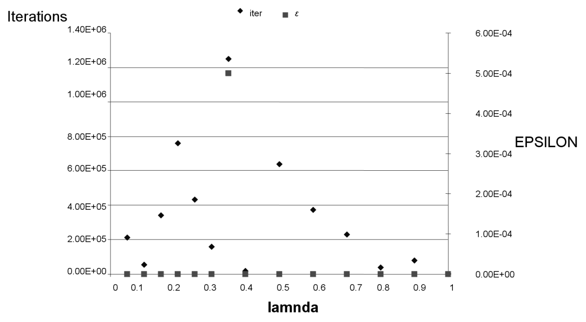

Fig. 4. Instance of the almost strongly correlated problem.

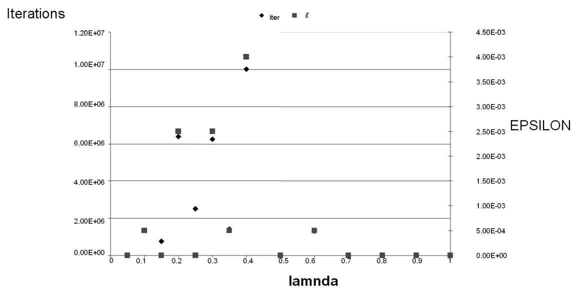

Fig. 5. Instance of the uncorrelated with similar weights problem.

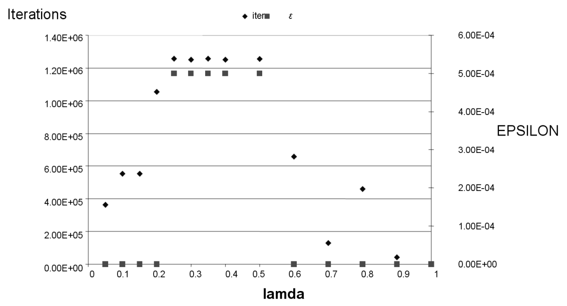

Fig. 6. Instance of the weakly correlated problem.

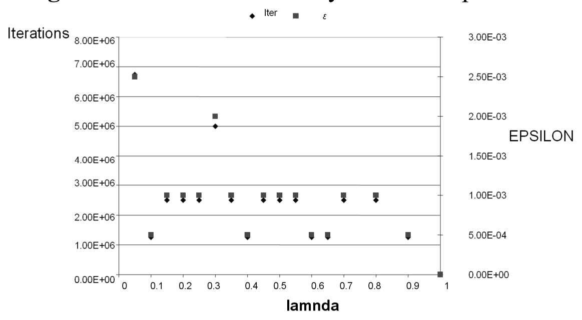

Fig. 7. Instance of the strongly correlated problem.

It can be observed in Figures 2 and 3 that in the instances of uncorrelated and subset sum problems, the exact solutions are obtained along the whole parameter λ. A minor complexity is apparent during the transition phase, during which the most complex instances are presented but also resolved. This occurs in the interval *$0.2 \leq \lambda \leq 0.3$*.

In the instance of the almost strongly correlated problem, Figure 4, in a single occasion an approximate solution was calculated, where only one increase of ε was necessary. In this case, the transition phase was between *$0.3 \leq \lambda \leq 0.4$*.

In the instance of the uncorrelated with similar weights problem, Figure 5, for six of the fifteen points an approximate solution was calculated, and for three of those points more than one increase of ε was required. The interval where the most complex points (the transition phase) existed was *$0.2 \leq \lambda \leq 0.4$*.

In the instance of the weakly correlated problem, Figure 6, for five of the fifteen instances an approximate solution was calculated with only one increment of ε was necessary. These points came in the transition phase between *$0.2 \leq \lambda \leq 0.6$*.

In the instances of the strongly correlated and inverse strongly correlated problems, Figures 7 and 8 respectively, the calculation of the approximate solution in all points of the parameter λ was required.

It can be observed that the ZC algorithm is sensitive to the complexity of the problems previous described. It can also be seen that in the sequence of graphs in Figures 2 through 8, the number of points with an approximate solution gradually increases, when the number of iterations increases, the complexity of the instances also grows.

It can be demonstrated that the transition phase comes before the parameter λ reaches the value of 0.5 for the following cases: uncorrelated, subset sum, almost strongly correlated, uncorrelated with similar weights, and weakly correlated. In the strongly correlated and inverse strongly correlated cases, this phase is not distinguished, as the solving of all instances is complex. The above-mentioned indicates that it is not certain that the complex instances most representative of the problem are being calculated when $\lambda = 0.5$.

Martello et, al., [6] mentions that the less complex instances, for example the uncorrelated, are those that come in practical problems. It is for these instances that the ZC algorithm found an exact solution.

For the strongly correlated instance, Table 1 shows the interval in which the optimum solution is found.

Table 1. Variation of the objective function for the instance of the strongly correlated problem

λ	ε	**Interval**	λ	ε	**Interval**
0.05	0.0005	$23{,}473 \leq z < 23{,}532$	0.50	0.0010	$160{,}970 \leq z < 161{,}130$
0.10	0.0005	$40{,}862 \leq z < 40{,}882$	0.55	0.0010	$175{,}215 \leq z < 175{,}390$
0.15	0.0010	$57{,}122 \leq z < 57{,}179$	0.60	0.0005	$189{,}520 \leq z < 189{,}614$
0.20	0.0010	$72{,}689 \leq z < 72{,}761$	0.65	0.0005	$203{,}705 \leq z < 203{,}806$
0.25	0.0010	$87{,}891 \leq z < 87{,}978$	0.70	0.0010	$217{,}730 \leq z < 217{,}947$
0.30	0.0020	$102{,}669 \leq z < 102{,}874$	0.80	0.0010	$245{,}780 \leq z < 246{,}025$
0.35	0.0010	$117{,}470 \leq z < 117{,}587$	0.90	0.0005	$273{,}841 \leq z < 273{,}977$
0.40	0.0005	$132{,}119 \leq z < 132{,}185$	1.00	0.0000	$z = 301{,}510$
0.45	0.0010	$146{,}503 \leq z < 146{,}649$			

As can be observed in Table 1, for the instances of the strongly correlated problem, $\bar{\varepsilon} = 0.001$ is obtained with a standard deviation of $\sigma = 0.0005$, which gives a prediction interval of $\varepsilon \pm \sigma$ between [0.0005, 0.0015]. For the uncorrelated and subset sum cases the exact solution is obtained along the whole parameter λ.

The ZC algorithm could be adjusted to obtain solutions with lesser values of kn^2, but a value close to the solution would be found in almost all the instances. Another option would be to allow the algorithm to continue iterations until the exact solutions of the complex instances are determined. This is possible but not practical, because in order to calculate instances of kn^5 with $m = 500$, iterations with an order of magnitude of 10^{13} would be required, while with kn^2 iterations, an order of magnitude of 10^5 iterations are all that is required.

4. Conclusions

It can be concluded that with the modifications to others' previous work made by the authors, the ZC algorithm that was elaborated was able to solve all the instances of the knapsack problem with kn^2 number of iterations.

Upon solving each problem with a parameter, it was demonstrated that even in the instances considered less complex, there is a transition phase. The ZC algorithm was able to calculate the approximate optimum solution with a complexity $O(n^2)$ in the transition phase.

The proposed algorithm is flexible in determining approximate solutions for the most complex instances, where the parameters *limit_iteration* and ε can be adjusted.

Finally, it was demonstrated that the transition phase depends on the problem that is solved. For less complex problems, the transition phase covers a small interval within the parameter λ. As the problems increase in complexity, this interval increases until it covers the entire domain for the most complex problems. In the uncorrelated and weakly correlated problems, the transition phase is always found when the values of the parameter are less than 0.5.

5. References

[1] S. Martello and P. Toth "Knapsack Problems, Algorithms and Computer Implementations" John Wiley & Sons, England 1990.

[2] S. Kirkpatrick, C. D. Gelatt Jr., and M. P. Vecchi, Optimization by simulated annealing, Science, Vol. 220, No. 4598, 13 May, 671-680, 1983.

[3] G. L. Nemhauser, L. A. Wolsey, "Integer and Combinatorial Optimization" A Wiley-Interscience Publication, John Wiley and Sons Inc., New York 1999

[4] I. P. Gent, P. Prosser, "The Constrainedness of Search" University of Strathclyde, Scotland 1999.

[5] T. Hogg "Which Search Problems Are Random?" Proceedings AAAI98, pp 438-443, 1998.

[6] Silvano Martello, David Pisinger, Paolo Toht, "New trends in exact algorithms for the 0-1 knaspsack problem" Technical Report 97/10, DIKU, University of Copenhagen, Denmark, 1997.

[7] G..B. Dantzig, Linear Programming and extensions, ISBN: 0-691-08000-3, Princeton university press, U.S.A, 1993.

[8] M. A. Cruz-Chávez, J. Frausto-Solís, Simulated Annealing with Restart to Job Shop Scheduling Problem Using Upper Bounds, LNAI, Springer Verlag Pub., ISSN: 0302-9743, Vol. 3070, pp. 860 - 865, 2004.

[9] M. A. Cruz-Chávez, Cooperative Processes for the Job Shop Scheduling Problem Using Simulated Annealing, PhD Thesis, ITESM, México, 2004. (In Spanish)

[10] M. E. Gómez-Torres, J. C. Zavala-Díaz, M. A. Cruz-Chávez, Spaces of search in a binary tree to solve discreet optimization problems, Proceedings of 5to Congress of Computation AGECOMP, UAEM, ISBN 968-878-273-4, 21-23 November, México, 2006. (In Spanish)

[10] R. Beber, B. Vöcking "Random Knapsack in Expected Polynomial Time" Max-Planck-Institut für Informatik Saarbrücken, Germany, 2003.

[11] J. Crispin Zavala-Díaz, V. Khachaturov, Integer programming, the tree of cubes method, their parallel algorithm and their applications, Contexts in the investigation of the social and administrative sciences, Autonomous University of Morelos State, First edition 2006. (In Spanish)