Efficient Implementations of Two Variant Subset Sum Problems — A Case Study of How to Process Appraisal Books Resulting from Fire-Destroyed Money

PeiZong Lee†, Fang-Yu Huang‡, Chorng-Yuan Huang§¶, and Hwann-Tzong Chen!

†Institute of Information Science, Academia Sinica, Taiwan, R.O.C.

‡National Bureau of Standard, Taipei, Taiwan, R.O.C.

§Dept. of EE, Yuan-Ze Inst. of Tech, Chung-Li, Taiwan, R.O.C.

¶Issuance Dept., The Bank of Taiwan, Taipei, Taiwan, R.O.C.

! Dept. of CSIE, National Taiwan Univ., Taipei, Taiwan, R.O.C.

leepe@iis.sinica.edu.tw

Key Words: subset sum; bucket sort; first fit; spiral round robin; object-oriented concept.

Abstract

This paper is concerned with designing efficient algorithms to process the appraisal books, which are used to estimate the amount of money lost due to a fire, in the Bank of Taiwan. First, we propose heuristic algorithms to pack appraisal books daily into bundles in the units of 1,000,000 dollars, 500,000 dollars, and 100,000 dollars. Second, we present efficient algorithms to take out certain appraisal books so that the amount of the remaining accumulated appraisal books is a multiple of one thousand dollars. We also introduce a software tool which integrates the functions of processing appraisal books. Experimental studies show that the software tool is very helpful for the clerks.

1 Introduction

This paper is concerned with designing efficient algorithms which will allow us to process the appraisal books which are used to estimate the amount of money lost due to a fire. (These books are stored in the Bank of Taiwan's head office.) This process involves two variant subset sum problems.

1.1 Appraisal Books of Burnt Paper Money

We first illustrate whence appraisal books of burnt paper money come. Burnt paper money means that paper money was burnt due to an accidental fire. In order to reduce the victim's losses, the victim can bring the ashes of the burnt paper money to a branch of the Investigation Bureau. After three weeks, the victim can get an appraisal book which indicates how much paper money this victim lost. Then, this victim can be reimbursed the same amount of money indicated in the appraisal book from any branch of the Bank of Taiwan. Therefore, from the Bank's point of view, appraisal books of burnt paper money are money although they cannot be used as hard currency.

To be presented in the *ACM Symposium on Applied Computing*, Philadelphia, Pennsylvania, February 18–20, 1996.

We now introduce how to process these appraisal books in the Bank of Taiwan. Each day, each branch of the Bank collects appraisal books and sends them to the head office of the Bank. They have the following rules to pack paper money. First, 100 pieces of the same plate paper money are packed as a pack. Then, 10 packs of the same plate paper money are packed as a bundle. Therefore, for the same plate 1000-face-value (500-face-value, or 100-face-value) paper money, each bundle contains 1,000,000 dollars (500,000 dollars, or 100,000 dollars, respectively). Since appraisal books are an uncommon occurrence, they are packed with other paper money. For instance, if appraisal books are packed within a bundle whose face value is 1000-dollars (500-dollars, or 100-dollars), then including the appraisal books and certain change, the total amount of the bundle is still 1,000,000 dollars (500,000 dollars, or 100,000 dollars, respectively).

We now state how the clerks process these appraisal books in the head office. The clerks have to collate all bundles of paper money. Paper money in good condition is packed together and will be circulated again. Worn-out paper money and appraisal books are also individually packed together; however, they will be destroyed independently someday. Because of a custom of the Bank, all the paper money and appraisal books still must be packed in the unit of a bundle of either 1,000,000 dollars or 500,000 dollars or 100,000 dollars. Each appraisal book is processed until it has been stamped by 16 official or personal seals.

The final step of destroying appraisal books requires another long-term official document. The Bank will invite persons from the Ministry of Finance, the Ministry of Audit, and the Central Bank of (the Republic of) China to investigate these appraisal books again and then to destroy these appraisal books. Because of these tedious bureaucratic rules, the Bank will only apply to destroy appraisal books once or twice each year.

1.2 Two Variant Subset Sum Problems

A formal definition of a subset sum problem is as follows [4]. *Instance: Given a finite set A, where size $s(a) \in Z^+$ for each $a \in A$, and a positive integer B. Question: Is there a subset $A' \subseteq A$ such that the sum of the sizes of the elements in A' is exactly B?* The process of packing appraisal books in the head office raises two variant subset sum problems. First, each day the clerks will receive some bundles of paper money from other branches which contain certain appraisal books. If a bundle contains more than one type of the same plate paper money or if it contains both paper money and appraisal books, then this bundle is not pure. They call this bundle a *dirty* bundle. It is another Bank's custom to avoid dirty bundles, because the clerks must handle these dirty bundles again someday. Therefore, the clerks will try to pack these newly arrived appraisal books with existing ones.

It is the clerks' intention to keep as few dirty bundles as possible. Because if a bundle is pure, then they can transfer this bundle to another department. However, if the bundle is dirty, they must keep this bundle. Because this dirty bundle has been stamped by their personal seals, they have a duty to saveguard this bundle before it can be transferred to another department. It is thus an imperative for the clerks to monitor closely bundles of paper money which are stamped by their personal seals. That is the reason why the clerks will try to pack appraisal books each day. The first subset sum problem is how to pack appraisal books without or with certain small changes in the unit of a bundle of either 1,000,000 dollars or 500,000 dollars or 100,000 dollars. In addition, both the number of bundles and the amount of paper money which exist are as small as possible.

Second, at the time of preparing to destroy accumulated appraisal books, the clerks will pack all appraisal books into one large bundle. However, according to a rule that each amount of the same plate of burnt paper money must be a multiple of one thousand dollars, therefore, the clerks have to take out some appraisal books so that the rule can be achieved. Because an appraisal book may contain both 100-face-value and 500-face-value burnt paper money, to take out one appraisal book may result in changing the amounts of both 100-face-value and 500-face-value burnt paper money. The second subset sum problem is how to take out certain appraisal books so that each total amount of the same plate of burnt paper money indicated in the remaining appraisal books is a multiple of one thousand dollars. In addition, the following constraints are optional: (1) the document ID numbers of appraisal books which will be taken out are as late as possible; (2) the number of appraisal books which will be taken out is as small as possible; (3) the total amount of burnt paper money represented by these appraisal books which will be taken out is as small as possible; and (4) the total amount of Hua-two-plate 100-face-value burnt paper money is as small as possible. Note that, "Hua" is the system used to describe Taiwan currency. This is because Hua-two-plate 100-face-value paper money will not be printed again, and the Bank would like to destroy them as early as possible. Note that, the last three constraints are perhaps inconsistent.

The general subset sum problem is NP-complete [4]. Although it can be solved by applying a dynamic programming algorithm in pseudo-polynomial time (it can be solved in time polynomial with respect to a parameter which is itself exponential in input size [4]), the algorithm requires to construct a table of size $m \times B$, where m is the number of objects in the set A, and B is the target positive integer. Therefore, for the first variant subset sum problem, if we are required to pack appraisal books into bundles of exactly 1,000,000 dollars or 500,000 dollars or 100,000 dollars, then the algorithm requires a large memory space for constructing the mentioned table, because 1,000,000 and 500,000 and 100,000 are all large numbers. Therefore, in practice, we only try to pack appraisal books as close to 1,000,000 dollars or 500,000 dollars or 100,000 dollars as possible. On the other hand, if the target value is a small integer, then the subset sum problem might be solved using a dynamic programming algorithm in pseudo-polynomial time. In our application, the second variant subset sum problem belongs to this class. However, since we require additional constraints for the solutions, we found that it is better to adopt a new data structure for the second variant subset sum problem.

The rest of this paper is organized as follows. In Section 2, we describe a software tool which we developed to assist the clerks in packing appraisal books. In Section 3, we present efficient algorithms for dealing with the two above-mentioned variant subset sum problems. In Section 4, some experimental studies using the software tool are reported. Finally, some concluding remarks are given in Section 5.

2 A Software Tool for Packing Appraisal Books

In this section, we introduce a software tool — PAT, which is designed for assisting the clerks in packing appraisal books. PAT is from the acronym of "Packing Appraisal books Tool". As mentioned in Section 1, the clerks receive several appraisal books each day. They will translate each appraisal book as a record and append it at the end of an ASCII file. As shown in Fig. 1 (at the end of this paper), each record contains seven items in sequence: (1) sequence number (NO.), (2) document ID, (3) applicant name (NAME), (4) amount of Hua-two-plate 100-face-value burnt paper money ($h2\_100$), (5) amount of Hua-three-plate 100-face-value burnt paper money ($h3\_100$), (6) amount of Hua-two-plate 500-face-value burnt paper money ($h2\_500$), and (7) amount of Hua-two-plate 1000-face-value burnt paper money ($h2\_1000$). PAT provides three main functions as shown in Fig. 2.

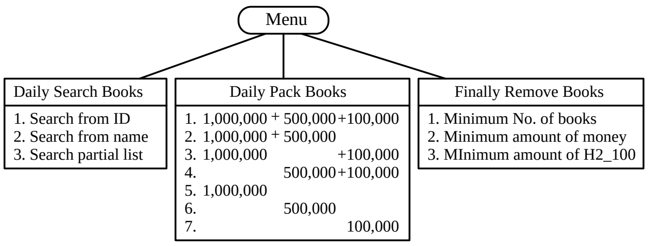

Figure 2: PAT provides three main functions.

First, *Daily Search Books*: this function allows the clerks to seek records based on a document ID or an applicant name. It also can search a partial list and sum them up with respect to each record and each total amount of $h2\_100$, $h3\_100$, $h2\_500$, and $h2\_1000$, individually. Second, *Daily Pack Books*: this function provides the clerks some suggestions to pack appraisal books daily. It allows the clerks to pack appraisal books into bundles in the units of either 1,000,000 dollars or 500,000 dollars or 100,000 dollars. Third, *Finally Remove Books*: this function is used when the clerks prepare to destroy accumulated appraisal books. As mentioned in Section 1, the clerks have to take out certain appraisal books so that each total amount of $h2\_100$, $h3\_100$, $h2\_500$, and $h2\_1000$ in the remaining accumulated appraisal books is a multiple of one thousand dollars. This

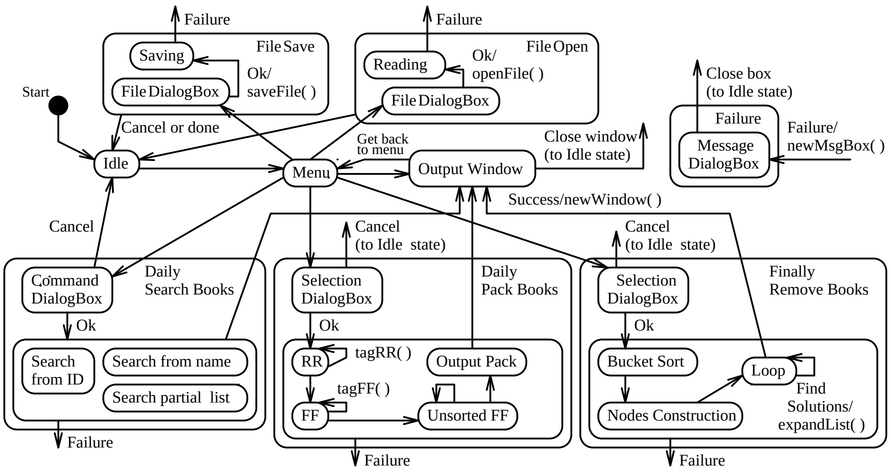

Figure 3: The state transition diagram of PAT.

function allows the clerks to select one of the following three constraints for the removed appraisal books: (1) a minimum number of appraisal books, (2) a minimum amount of money, and (3) a minimum amount of $H2\_100$. Note that, we use $H2\_100$ here to represent the total amount of $h2\_100$ in several appraisal books. The third constraint is because Hua-two-plate 100-face-value paper money will not be printed again, and the Bank would like to destroy them as early as possible.

2.1 The Implementation of PAT

PAT is designed using object-oriented concepts [2] and is implemented using Borland C++ [1] [5] and can be run on PCs. Fig. 3 shows the state transition diagram of PAT. In this system, all of the main menus, windows, dialogboxes (which are scroll-bar window menus for selecting functions or for accepting commands), and data records are objects. Users can change the current state or data values by using the provided member functions.

We now give a detailed description of the system. In the main menu, users can select the following six functions: (1) search appraisal books daily (Daily Search Books), (2) pack appraisal books daily (Daily Pack Books), (3) pack appraisal books finally (Finally Remove Books), (4) open a file (FileOpen), (5) save a file (FileSave), and (6) adjust output window (Output Window). Function 4 which opens a file, Function 5 which saves a file, and Function 6 which adjusts output windows can be implemented by using conventional file-open functions, write functions, and window-library functions, respectively. Function 1 provides users three subfunctions to search appraisal books daily. All of these three subfunctions: search from an ID, search from one name, and search a partial list can be implemented using sequential-search techniques. Function 2 and Function 3, which deal with the core applications of packing appraisal books daily and packing appraisal books finally, will be discussed in the next section.

3 Algorithms for Packing Appraisal Books

3.1 Heuristic Algorithms for Packing Appraisal Books Daily

Function 2 provides users suggestions for packing appraisal books daily. It allows users to pack appraisal books into bundles which are close to the combinations of 1,000,000 dollars and 500,000 dollars and 100,000 dollars. As mentioned in Section 1, the problem of packing a subset of appraisal books into an exact and large amount of money such as 1,000,000 dollars or 500,000 dollars or 100,000 dollars is as difficult as a general subset sum problem. Therefore, we adopt heuristic algorithms for packing appraisal books into bundles which are as close to 1,000,000 dollars or 500,000 dollars or 100,000 dollars as possible.

We propose three algorithms for daily packing of appraisal books, and evaluate their performances using a least-square method. All three of these algorithms will pack bundles for 1,000,000 dollars first, then will pack bundles for 500,000 dollars, and then will pack bundles for 100,000 dollars. The first algorithm contains two steps. First, it sorts the amount of each appraisal books in a decreasing order using a bucket sort algorithm [3] [6]. Then, it adopts a *spiral round-robin* (RR) strategy to distribute appraisal books into different bundles. Now, suppose that we want to pack appraisal books into N bundles.

The first algorithm:

Step 1: sort the amount of each appraisal book in a decreasing order using a bucket sort algorithm;

Step 2: /* Spiral round-robin algorithm. */
consider each appraisal book one by one in the sorted sequence:

sequentially assign one appraisal book form Bundle 1 to Bundle N, then from Bundle N to Bundle 1, in a zigzag fashion. If an appraisal book cannot be packed into any bundle, then pack it into the remaining bundle.

The second algorithm also contains two steps. First, it also sorts the amount of each appraisal book in a decreasing

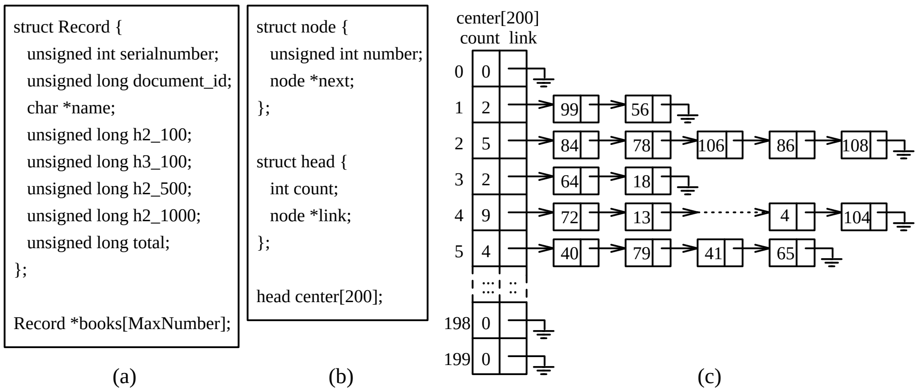

Figure 4: The data structures of PAT.

order the same as the first algorithm. Second, it employs a *first-fit* (FF) policy to distribute appraisal books into different bundles.

The second algorithm:

Step 1: sort the amount of each appraisal book in a decreasing order using a bucket sort algorithm;

Step 2: /* First-fit algorithm. Note that, using this policy, the last bundle is the remaining bundle. */
consider each appraisal book one by one in the sorted sequence:

sequentially assign appraisal books one by one into Bundle i for $i \geq 1$. If an appraisal book cannot be packed into Bundle i, then it will be packed into Bundle j later, where $j > i$.

Unlike the first two algorithms which are required to sort the amount of each appraisal book, the third algorithm does not use any sorting function. It only applies the first-fit policy to the unsorted sequence of appraisal books. Therefore, the third algorithm is equivalent to the second step of the second algorithm, except that it deals with an unsorted sequence of appraisal books. Note that, the complexity of the first two algorithms is bounded by the bucket sort algorithm which is linear, and the complexity of the third algorithm is also linear.

We now introduce the least-square formula which we use to evaluate the performances of the above three algorithms. Suppose that appraisal books are packed into N_1 bundles of 1,000,000 dollars, N_2 bundles of 500,000 dollars, N_3 bundles of 100,000 dollars, and one remaining bundle. Then the least-square formula is as follows:

$10*\sum_{i=1}^{N_1}(1,000,000-\text{total amount of Bundle } i)^2+5*\sum_{j=1}^{N_2}(500,000-\text{total amount of Bundle } j)^2+\sum_{k=1}^{N_3}(100,000-\text{total amount of Bundle } k)^2$.

3.2 Efficient Algorithms for Packing Appraisal Book Finally

Function 3 provides users suggestions for packing appraisal books finally at the time of preparing to destroy accumulated appraisal books. As mentioned in Section 1, according to a rule in the Bank, each total amount of $h2\_100$, $h3\_100$, $h2\_500$, and $h2\_1000$ in the appraisal books must be a multiple of one thousand dollars. Therefore, if the total amount of either $h2\_100$ or $h3\_100$ or $h2\_500$ has a few change of several hundred dollars, the clerks have to take out some appraisal books so that the rule can be achieved.

For instance, for the sample example of the 108 appraisal books in Fig. 1, each total amount of $h2\_100$, $h3\_100$, and $h2\_500$ is 14,300 dollars, 128,900 dollars, and 123,000 dollars, respectively. If we take out No. 16 and No. 23 and No. 40 three appraisal books, then each total amount of $h2\_100$, $h3\_100$, and $h2\_500$ in the remaining accumulated appraisal books is 12,000 dollars, 127,000 dollars, and 122,000 dollars, respectively. Therefore, to take out No. 16 and No. 23 and No. 40 appraisal books is one solution.

Function 3 allows users to select one of the following three options for the removed appraisal books: (1) minimum number of appraisal books, (2) minimum amount of money, and (3) minimum amount of $H2\_100$. In addition, Function 3 also requires the solution appraisal books to have the latest document ID. A naive implementation of Function 3 can do an exhaustive search from all of the combinations of t records (appraisal books), for $t \geq 0$. Then, the searching complexity is $O(\sum_{i=1}^{t}(i * C_i^m))$, where m is the number of appraisal books. This *ad hoc* implementation certainly will not be acceptable when the number of appraisal books is large (due to requiring a large computation time).

In the following, we present an efficient algorithm for implementing Function 3. We first introduce our data structures. Fig. 4-(a) shows the data structure of each record which represents an appraisal book. We now consider the cases when $h2\_100$, $h3\_100$, and $h2\_500$ are not multiples of one thousand. It is true that $0 \leq (h2\_100/100) \bmod 10 \leq 9$, $0 \leq (h3\_100/100) \bmod 10 \leq 9$, and $0 \leq (h2\_500/500) \bmod 2 \leq 1$. Therefore, it is natural to construct an array of pointers of size 200 to record each appraisal book. Fig. 4-(b) shows the data structure of the array of pointers $center[200]$. Each element of the array has two fields: (1) count, which represents the numbers of records; and (2) link, which points to a node, and the node represents one record.

If $i = (h2\_100/100) \bmod 10$, $j = (h3\_100/100) \bmod 10$, and $k = (h2\_500/500) \bmod 2$ for the record r, then record r will be stored in a node which then will be inserted in the head of the linked list pointed by $center[h].link$, where $h =$

$i*10*2+j*2+k$. This is the row-major transformation from a three-dimensional index (i, j, k) to a one-dimensional index h with the basis $10 \times 10 \times 2$. For convenience, we define the following two functions to do the row-major transformation and the inverse row-major transformation. $m2a$ means from a matrix to an array; $a2m$ means from an array to a matrix.

$$m2a(i, j, k) = i * 10 * 2 + j * 2 + k;$$
$$a2m(h) = (\lfloor h/20 \rfloor, \lfloor (h \bmod 20)/2 \rfloor, h \bmod 2).$$

In order to guarantee that the solution appraisal books have the latest document IDs and satisfy one of the three options, before storing records into the array of points, we apply a bucket sort algorithm three times with respect to keys from document IDs, the amount of each record, and $h2\_100$ in each record in a certain order. If we choose the first option, we only need to sort all records according to document IDs in an ascending order. If we choose the second option, we first sort all records according to document IDs in an ascending order, then we sort all records according to $h2\_100$ in each record in a descending order, and then we sort all records again according to the amount of each record in a descending order. If we choose the third option, we also sort all records according to document IDs in an ascending order first, but then we sort all records according to the amount of each record in a descending order, and then we sort all records again according to $h2\_100$ in each record in a descending order. Therefore, the record numbers of the best solution which is composed of t appraisal books will appear in the first node of the t corresponding linked lists. Fig. 4-(c) shows parts of the array of pointers and linked lists, which are constructed from our sample example in Fig. 1, when we choose the second option. Note that, for the case when $i = 0$, $j = 0$, and $k = 0$ which imply $h = 0$ for some record r, we will not store record r into the linked list pointed by $center[0]$, as record r has no influence for any solutions.

After constructing our data structure, we can find solutions as follows. Suppose that each total amount of $h2\_100$, $h3\_100$, and $h2\_500$ in all records are $H2\_100$, $H3\_100$, and $H2\_500$, respectively. Let

$$i = (H2\_100/100) \bmod 10;$$
$$j = (H3\_100/100) \bmod 10;$$
$$k = (H2\_500/500) \bmod 2.$$

Then, we can check whether there exists one solution which comprises t appraisal books for $t \geq 0$ as follows. If $m2a(i, j, k) = 0$, then each value of $H2\_100$, $H3\_100$, and $H2\_500$ is a multiple of one thousand dollars and we need not to take out any appraisal book. In the above case, $t = 0$.

If $m2a(i, j, k) = h$ and $center[h].count \neq 0$, then each solution contains only one appraisal book, and the solution with the latest document ID is the first node in the linked list pointed by $center[h].link$. In the above second case, $t = 1$. Otherwise, if a solution comprises $t > 1$ records, the procedure is shown in Fig. 5. In addition, the solution, which is with the latest document ID, comprises each first node in the corresponding t linked lists. The rationale behind the algorithm can be formulated as an algebraic equation as follows: Given a basis y, a target value $0 \leq x < y$, and $t-1$ integers x_1, x_2, ..., x_{t-1}, then there exists exactly one integer $0 \leq x_t < y$ such that $x_1 + x_2 + \cdots + x_{t-1} + x_t \bmod y = x$.

for $h_1 \in \{h \mid center[h].count \neq 0\}$
  for $h_2 \in \{h \mid center[h].count \neq 0$ and $h \geq h_1\}$
    $\vdots$
   for $h_{t-1} \in \{h \mid center[h].count \neq 0$ and $h \geq h_{t-2}\}$ {
    $(i_1, j_1, k_1) = a2m(h_1)$;
    $(i_2, j_2, k_2) = a2m(h_2)$;
    $\vdots$
    $(i_{t-1}, j_{t-1}, k_{t-1}) = a2m(h_{t-1})$;
    $i_t = (i - i_1 - i_2 - \cdots - i_{t-1}) \bmod 10$;
    $j_t = (j - j_1 - j_2 - \cdots - j_{t-1}) \bmod 10$;
    $k_t = (k - k_1 - k_2 - \cdots - k_{t-1}) \bmod 2$;
    $h_t = m2a(i_t, j_t, k_t)$;
    if ($h_t < h_{t-1}$ or $center[h_t].count = 0$) **then continue**;
    if ($check\_replicated\_entries$ = **FALSE**) **then continue**;
    list all solutions, and the solution which comprises the first node of each corresponding linked list is the best solution.
   }

function $check\_replicated\_entries$ {
  $F(t) = 1$;
  for $s = t - 1$ down to 1
    if ($h_s = h_{s+1}$) **then** $F(s) = F(s+1) + 1$;
    else $F(s) = 1$;
  $s = 1$;
  while ($s \leq t$) {
    if ($center[h_s].count < F(s)$) **then return FALSE**;
    $s = s + F(s)$; }
  return TRUE.
}

Figure 5: The procedure of checking whether a solution comprises t records.

We now analyze the computation complexity. Suppose that there are m' nonnull entries in the array of pointers of $center[200]$. Then the above computation complexity of finding one solution which comprises t appraisal books is $O(\sum_{i=1}^{t}(i * C_{i-1}^{m'}))$. This complexity is at least one order better than the one using an *ad hoc* search algorithm which requires $O(\sum_{i=1}^{t}(i*C_i^m))$ computation complexity, where m is the number of appraisal books and $m' \leq m$. For instance, for our sample example which contains 108 appraisal books in Fig. 1, $m = 108$ and $m' = 23$. Let $t = 3$, then $O(\sum_{i=1}^{3}(i* C_{i-1}^{23}))$ is at least one order less than $O(\sum_{i=1}^{3}(i * C_i^{108}))$.

4 Experimental Studies

In this section, we present experimental studies of the software tool. We study three examples. The first one is our sample example which contains 108 appraisal books of the 1994 fiscal year in Fig. 1. The second example contains 86 appraisal books of the 1995 fiscal year in Fig. 6. The third one contains 189 appraisal books which are the union of those in the first example and in the second example. Note that, the first five records in Fig. 6 are identical to five records: No. 16, No. 105, No. 106, No. 107, and No. 108 in Fig. 1. Table 1 shows experimental studies for Function 2 (which packs appraisal books daily) and Function 3 (which packs appraisal books finally). For Function 2, appraisal books are packed into bundles in the units of all 1,000,000 dollars, 500,000 dollars, and 100,000 dollars. The term 4-1-1 means to pack appraisal books into four bundles of 1,000,000 dollars, one bundle of 500,000 dollars, one bundle of 100,000 dollars, and one remaining bundle.

Cases	1. 108 records	2. 86 records	3. 189 records
$H2\_100$	14300	7300	19300
$H3\_100$	128900	201000	328000
$H2\_500$	123000	124000	245000
$H2\_1000$	4382000	3710000	7944000
Total amount	4648200	4042300	8536300
Function 2: Daily pack			
Sorted RR			
# of bundles	4–1–1	4–0–0	8–1–0
remaining bundle	49900	44500	38700
least square root	3068	4025	3369
Sorted FF			
# of bundles	4–1–1	4–0–0	8–1–0
remaining bundle	49800	43600	38500
least square root	2581	2950	3122
Unsorted FF			
# of bundles	4–1–1	4–0–0	8–1–0
remaining bundle	76200	216000	216000
least square root	32610	535440	395134
Function 3: Final pack			
# of suggestions	7	15	24
Minimum # of records	3	1	1
Record No.(s)	16, 104, 107	1	16
Minimum amount of money	50200	43300	36300
Record No.(s)	16, 23, 40	1	80, 147
Minimum amount of $H2\_100$	2300	300	300
Record No.(s)	16, 23, 40	6, 30, 54	59, 157

Table 1: Experimental studies of Function 2 and Function 3 for three examples. The unit of money is 'dollar'.

In Table 1, we can find that for Function 2, the performance of the sorted first-fit algorithm is the best and the performance of the unsorted first-fit algorithm is the worst in these three examples. For Function 3, if the option we choose is the minimum number of records to take out, then records with the latest document IDs will be reported. Furthermore, the solutions of choosing different options may result in different answers.

It is also worthwhile to mention that the software tool is very helpful for the clerks in the Bank. In 1994, the clerks still had to pack appraisal books themselves. When approaching the preparation time to destroy accumulated appraisal books, for the case of the 108 appraisal books in 1994 fiscal year in Fig. 1, the clerks took out five appraisal books: No. 16, No. 105, No. 106, No. 107, and No. 108. The total amount of these five appraisal books is 154,200 dollars which includes 2,300 dollars for the Hua_two_plate 100_face_value burnt paper money. On the other hand, as shown in Table 1, the software tool suggests taking out only three appraisal books, and the total amount of these three appraisal books is only 50,200 dollars which includes 2,300 dollars for the Hua_two_plate 100_face_value burnt paper money. Obviously, the software tool has beaten the clerks in terms of efficiency.

5 Conclusions

We have presented in this paper a software tool for assisting the clerks in the Bank of Taiwan in packing appraisal books of burnt paper money. This software tool is designed using object-oriented concepts and is implemented using Borland C++ and can be run on PCs. The main algorithms are proposed to solve two variant subset sum problems, in which the first one is used to solve the daily packing of appraisal books and the second one is used to solve the final packing of appraisal books. The other program parts are designed for a friendly user interface environment. Currently, this software tool is available on an on-line service. According to the responses from the clerks, this software tool is very helpful, and they can now easily finish all of the packings based on the suggestions derived from the software tool before the time of closing of the treasury at 4:00pm each day.

Acknowledgements

This work was partially supported by the NSC under Grant NSC 85-2213-E-001-006. The authors would like to thank anonymous referees for their valuable comments on this paper.

References

[1] *Turbo C++ Version 3.0 User's Guide*. Borland International, Inc., Scotts Valley, CA, 1992.

[2] G. Booch. *Object-Oriented Analysis and Design with Applications*. The Benjamin/Cummings Publishing Company, Inc., Redwood City, CA, second edition, 1994.

[3] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. The MIT Press, Cambridge, MA, 1990.

[4] M. R. Garey and D. S. Johnson. *Computers and Intractability, A Guide to the Theory of NP-Completeness*. W. H. Freeman & Company, Publishers, San Francisco, CA, 1979.

[5] R. Sedgewick. *Algorithms in C++*. Addison-Wesley Publishing Company, Reading, MA, 1992.

[6] M. A. Weiss. *Data Structrues and Algorithm Analysis in C++*. The Benjamin/Cummings Publishing Company, Inc., Redwood City, CA, 1994.

NO.	DOCUMENT ID	NAME	H2_100	H3_100	H2_500	H2_1000
NO.1	(82)6934	H.-Y. Yeh	100	1800	0	23000
NO.2	(82)7001	F.-T. Yui	500	0	500	0
NO.3	(83)7052	G.-L. Liu	0	4700	9000	170000
NO.4	(82)7007	T.-Y. Yeh	0	200	0	50000
NO.5	(82)6983	T. Lee	0	0	0	14000
NO.6	(82)6947	A.-J. Xu	0	4400	500	44000
NO.7	(83)7081	H.-J. Wang	0	0	0	62000
NO.8	(83)7067	B.-S. Yu	6600	3800	0	107000
NO.9	(83)7080	H.-C. Wang	0	200	7000	17000
NO.10	(83)7074	Q.-T. Lee	0	4000	4000	20000
NO.11	(83)7039	C.-J. Chen	0	0	0	10000
NO.12	(83)7071	Y.-C. Lan	0	400	500	0
NO.13	(82)6989	C.-C. Liu	0	2200	0	3000
NO.14	(83)7076	C.-H. Chiu	0	0	0	140000
NO.15	(82)6977	L.-K. Wen	0	900	500	0
NO.16	(83)7050	Y.-B. Kuo	2300	0	0	41000
NO.17	(82)6991	H.-C. Chen	100	1900	1500	4000
NO.18	(83)7090	Y.-H. Yen	0	100	2500	102000
NO.19	(83)7085	C.-C. Tsai	0	300	0	29000
NO.20	(83)7041	H. Kao	0	0	0	11000
NO.21	(81)6876	S.-J. Shen	0	0	0	57000
NO.22	(83)7097	J.-K. Chen	0	500	500	29000
NO.23	(83)7086	S.-C. Horng	0	700	500	3000
NO.24	(83)7082	H.-C. Lin	0	0	0	20000
NO.25	(83)7098	C.-K. Chen	0	400	5500	31000
NO.26	(83)7109	G.-W. Huang	0	0	0	32000
NO.27	(83)7062	N.-S. Cheng	0	400	2000	59000
NO.28	(83)7107	Y.-H. Chen	0	0	0	13000
NO.29	(83)7088	C.-L. Kuo	0	200	0	30000
NO.30	(83)7118	T.-Y. Shieh	0	2800	0	18000
NO.31	(83)7121	C.-J. Cheng	0	3000	0	29000
NO.32	(83)7106	K.-S. Yao	0	0	0	31000
NO.33	(83)7117	L.-C. Wu	0	0	0	32000
NO.34	(83)7068	C.-S. Tung	0	16300	15000	0
NO.35	(83)7115	B.-Y. Tsai	0	400	500	39000
NO.36	(83)7104	L.-T. Teng	0	3300	8000	14000
NO.37	(83)7123	P.-W. Chen	0	0	0	58000
NO.38	(83)7057	Y.-C. Wang	0	400	1000	14000
NO.39	(82)6926	K.-H. Mao	0	0	0	44000
NO.40	(82)6967	Y.-H. Yang	0	1200	500	1000
NO.41	(83)7038	N.-Y. Shieh	0	200	500	82000
NO.42	(83)7126	T.-F. Wu	0	0	0	117000
NO.43	(83)7116	S.-T. Chou	0	0	0	143000
NO.44	(83)7056	M.-S. Chang	0	0	0	11000
NO.45	(83)7119	H.-S. Chien	0	0	0	84000
NO.46	(83)7133	J.-T. Chow	100	0	0	12000
NO.47	(83)7099	Y.-T. Dong	900	1100	2000	34000
NO.48	(83)7138	S.-Y. Hon	0	0	0	11000
NO.49	(83)7108	C.-T. Hung	0	0	0	162000
NO.50	(83)7092	S.-L. Kan	0	0	0	190000
NO.51	(83)7127	Y.-C. Lii	0	1500	3500	96000
NO.52	(83)7124	S.-T. Chou	0	0	0	50000
NO.53	(82)6988	C.-N. Yang	0	200	1000	8000
NO.54	(83)7027	H.-C. Wong	0	0	0	27000
NO.55	(82)6982	J.-H. Liu	0	300	0	12000
NO.56	(83)7091	Y.-C. Mou	0	5000	13500	25000
NO.57	(83)7084	K.-C. Sha	0	0	0	144000
NO.58	(82)6980	C.-H. Lin	0	3900	1500	34000
NO.59	(80)6680	H.-S. Maa	0	1500	0	0
NO.60	(82)6978	S.-Y. Wen	0	0	0	24000
NO.61	(83)7083	H.-H. Tso	100	400	2000	0
NO.62	(83)7087	K.-S. Shyu	0	400	1500	17000
NO.63	(83)7102	T.-P. Tsong	0	0	1000	50000
NO.64	(83)7063	C.-S. Lee	0	100	500	18000
NO.65	(83)7075	S.-W. Tse	0	5200	500	85000
NO.66	(83)7077	M.-T. Ho	0	0	0	115000
NO.67	(83)7114	K.-E. Lin	0	300	0	12000
NO.68	(83)7066	R.-N. Hwang	0	9400	2500	0
NO.69	(83)7111	J.-W. Liang	0	0	0	17000
NO.70	(83)7134	D.-Y. Yao	0	17800	9000	130000
NO.71	(83)7151	Y.-C. Tseng	0	700	0	24000
NO.72	(83)7135	C.-L. Chien	0	200	1000	4000
NO.73	(83)7153	W.-C. Wang	0	0	0	37000
NO.74	(83)7150	F.-L. Chang	0	0	0	112000
NO.75	(83)7149	K.-C. Hu	0	300	0	0
NO.76	(83)7155	B.-W. Yeung	0	0	0	11000
NO.77	(83)7148	C.-G. Kiang	0	4800	5000	13000
NO.78	(83)7073	H.-Y. Tzeng	0	2100	1000	6000
NO.79	(83)7146	U.-S. Jen	0	2200	500	17000
NO.80	(83)7120	L.-H. Yu	3600	1500	1500	22000
NO.81	(83)7145	P.-S. Li	0	1000	0	57000
NO.82	(83)7129	K.-C. Chan	0	0	0	52000
NO.83	(83)7140	Y.-T. Chen	0	0	0	52000
NO.84	(83)7161	K.-P. Teng	0	1100	0	4000
NO.85	(83)7166	C.-S. Lin	0	0	0	202000
NO.86	(83)7113	S.-B. Shiau	0	100	0	29000
NO.87	(83)7122	W. Liou	0	0	0	47000
NO.88	(83)7156	C.-G. Jon	0	0	0	21000
NO.89	(83)7096	Y.-C. Yu	0	0	0	58000
NO.90	(83)7130	M.-C. Wei	0	0	0	43000
NO.91	(83)7139	S.-C. Chang	0	200	0	36000
NO.92	(83)7152	K.-Y. Hwu	0	4200	10000	4000
NO.93	(83)7141	W.-K. To	0	500	1000	6000
NO.94	(83)7175	S.-I. Hwang	0	1000	0	42000
NO.95	(83)7164	Y.-H. Chao	0	0	0	14000
NO.96	(83)7163	P. Yeh	0	0	0	16000
NO.97	(83)7160	T.-Y. Wu	0	0	0	26000
NO.98	(83)7162	R.-Y. Shen	0	400	2000	6000
NO.99	(83)7174	R.-C. Hwang	0	0	500	11000
NO.100	(83)7169	J. Chen	0	0	0	14000
NO.101	(83)7178	C.-J. Yang	0	0	0	27000
NO.102	(83)7143	P.-H. Lee	0	4700	0	57000
NO.103	(83)7170	M.-S. Shih	0	0	0	7000
NO.104	(83)7172	L.-K. Kueh	0	200	0	87000
NO.105	(83)7171	M.-K. Hui	0	0	0	29000
NO.106	(83)7142	C.-D. Wu	0	100	2000	22000
NO.107	(83)7154	C.-C. Lin	0	1700	0	4000
NO.108	(83)7177	Y.-K. Fan	0	100	0	52000

Figure 1: The first 108 appraisal books in 1994 fiscal year.

NO.	DOCUMENT ID	NAME	H2_100	H3_100	H2_500	H2_1000
NO.1	(83)7050	Y.-B. Kuo	2300	0	0	41000
NO.2	(83)7171	M.-K. Hui	0	0	0	29000
NO.3	(83)7142	C.-D. Wu	0	100	2000	22000
NO.4	(83)7154	C.-C. Lin	0	1700	0	4000
NO.5	(83)7177	Y.-K. Fan	0	100	0	52000
NO.6	(83)7101	W.-C. Wu	0	300	0	8000
NO.7	(83)7125	C.-T. Lee	0	300	500	79000
NO.8	(83)7128	C.-Y. Chai	0	3000	14500	30000
NO.9	(84)7192	N.-H. Yeung	600	16200	7000	253000
NO.10	(83)7181	T.-C. Lin	0	0	0	20000
NO.11	(83)7183	S.-L. Kao	0	1200	500	108000
NO.12	(83)7157	Y.-L. Chau	0	100	500	6000
NO.13	(82)7144	S.-M. Ho	0	0	0	88000
NO.14	(83)7112	M.-J. Jiang	0	5000	7000	0
NO.15	(84)7193	H.-J. Tai	0	0	0	30000
NO.16	(83)7185	H.-W. Pan	0	0	0	51000
NO.17	(82)7167	K.-T. Tang	0	19100	500	18000
NO.18	(83)7190	H.-M. Tao	0	200	0	9000
NO.19	(83)7158	H.-H. Lo	0	0	0	139000
NO.20	(83)7095	L.-J Sung	0	0	0	32000
NO.21	(81)7137	H.-S. Ngai	0	700	3500	7000
NO.22	(83)7110	S.-H. Fang	0	3000	4500	36000
NO.23	(83)7089	T.-T. Kuo	0	0	0	18000
NO.24	(84)7201	F.-M. Tam	0	0	0	11000
NO.25	(83)7184	Y.-S. Ng	0	0	0	45000
NO.26	(83)7180	Z.-M. Lai	0	8200	10500	108000
NO.27	(83)7165	D. Hsiao	0	23000	0	32000
NO.28	(84)7198	Y.-T. Chiu	0	8800	0	28000
NO.29	(84)7202	C.-L. Song	0	1200	0	56000
NO.30	(84)7203	H.-C. Gow	0	1200	0	0
NO.31	(84)7199	P.-S. Tsai	0	3200	0	172000
NO.32	(83)7186	I.-S. Chan	0	0	0	50000
NO.33	(83)7093	C.-J. Su	0	0	500	40000
NO.34	(84)7204	I.-H. Tsai	0	9100	4500	25000
NO.35	(83)7189	B.-T. Lo	0	0	1000	23000
NO.36	(84)7208	S.-Y. Chang	0	7200	0	376000
NO.37	(84)7209	M.-F. Chen	0	0	0	28000
NO.38	(83)7103	M.-Y. Lee	0	500	500	0
NO.39	(84)7223	L.-Y. Chen	0	0	0	62000
NO.40	(84)7228	T.-G. Ongg	0	0	0	36000
NO.41	(84)7216	L.-S. Lin	0	0	4000	0
NO.42	(84)7232	J.-C. Chang	0	0	0	7000
NO.43	(84)7231	J.-M. Hseu	0	0	0	88000
NO.44	(84)7194	T.-S. Chu	700	1500	1500	4000
NO.45	(84)7219	F.-K. Huang	0	0	0	97000
NO.46	(84)7207	L.-J. Wu	0	0	0	24000
NO.47	(84)7217	H.-K. Chang	0	0	0	16000
NO.48	(84)7218	S.-Y. Chow	0	0	0	8000
NO.49	(84)7205	C.-S. Lee	0	5900	4500	12000
NO.50	(84)7229	G.-G. Chang	0	0	0	27000
NO.51	(84)7224	J.-R. Ho	0	2300	2500	41000
NO.52	(83)7187	Q.-R. Jan	0	0	0	16000
NO.53	(84)7245	Y.-C. Chang	0	0	0	17000
NO.54	(84)7195	H.-S. Gu	300	2500	0	37000
NO.55	(84)7236	C.-H. Lin	0	0	0	100000
NO.56	(84)7233	C.-S. Tsaur	0	0	0	17000
NO.57	(84)7244	K.-S. Wu	0	0	0	16000
NO.58	(84)7240	T.-M. Peng	0	0	0	61000
NO.59	(83)7159	S.-F. Chia	0	0	0	46000
NO.60	(84)7212	C.-I. Liao	0	1000	500	93000
NO.61	(84)7197	S.-C. Chen	0	16000	13500	0
NO.62	(84)7249	T.-T. Kuo	0	400	0	16000
NO.63	(84)7239	S.-N. Lin	0	300	500	66000
NO.64	(84)7257	F.-J. Shaw	0	600	1500	36000
NO.65	(84)7247	S.-J. Tu	3000	0	0	0
NO.66	(84)7254	M.-C. Chen	0	11700	1500	118000
NO.67	(84)7213	W.-Y. Yang	0	2400	12000	32000
NO.68	(84)7226	H.-C. Tsou	0	0	0	8000
NO.69	(84)7227	Y.-Z. Chen	0	0	0	10000
NO.70	(84)7235	N.-T. Wen	0	5000	0	23000
NO.71	(84)7251	Y.-C. Lin	0	4800	14000	43000
NO.72	(84)7211	H.-C. Chou	400	2700	1500	42000
NO.73	(84)7261	H.-F. Shu	0	3100	500	0
NO.74	(84)7250	Y.-K. Lo	0	1100	1500	14000
NO.75	(84)7260	C.-L. Han	0	1500	500	38000
NO.76	(84)7210	T.-H. Shee	0	0	0	3000
NO.77	(84)7242	L.-Y. Yung	0	20500	2000	0
NO.78	(84)7248	E.-R. Taam	0	3100	500	7000
NO.79	(84)7243	K.-H. Yee	0	0	0	3000
NO.80	(84)7214	S.-L. Lee	0	800	4000	11000
NO.81	(84)7221	J.-K. Chen	0	0	0	43000
NO.82	(84)7222	W.-D. Wang	0	0	0	17000
NO.83	(84)7269	H.-C. Chang	0	0	0	12000
NO.84	(84)7258	Y.-H. Liao	0	400	0	2000
NO.85	(84)7272	H.-W. Tsai	0	0	0	51000
NO.86	(84)7274	T.-D. Lee	0	0	0	216000

Figure 6: The first 86 appraisal books in 1995 fiscal year.