Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 6, Number 3, July 2022
DOI : https://doi.org/ 10.33395/sinkron.v7i3. 11592
e-ISSN : 2541-2019
p-ISSN : 2541-044X

Discrete Optimization Model in Constructing Optimal Decision Tree

Nurul Azri Azwar 1), **Parapat Gultom**2), **Sawaluddin**3)
University of Sumatera Utara, Medan, Indonesia
nurulazriazwar@gmail.com

Submitted : July 31, 2022 | **Accepted** : June 25, 2022 | **Published** : August 13, 2022

Abstract. Decision trees have been well studied and widely used in knowledge discovery and decision support systems. One of the applications of binary integer programming to form decision trees or decision making is the knapsack problem. The knapsack problem is an integer programming problem that involves only one constraint. The knapsack problem is generally illustrated with a bag and an item. The problem to be solved is to maximize the price of goods with a certain capacity that can be loaded by a bag with a certain capacity too. In solving the knapsack problem, it can generally be done directly. In this paper we are interested to show how the implicit enumeration method solves the knapsack problem to form an optimal decision tree.

Keywords: Decision tree, Optimization model, knapscak problem

INTRODUCTION

Decision tree approximate discrete-valued target function as trees and widely used practical method for inductive inference (Mitchell & Mitchell, 1997). Decision trees have prospered in knowledge discovery and decision support systems because of their natural and intuitive paradigm to classify a pattern through a sequence of questions. Decision trees have been a very popular class of predictive models for decades due to their interpretability and good performance on categorical features. Decision trees (DTs, for short) are similar to flow-charts as they apply a sequence of binary tests or decisions to predict the output label of the input data (Zhou, 2021).

One of the applications of binary integer programming to form decision trees or decision making is the knapsack problem. The knapsack problem is an integer programming problem that involves only one constraint (Martello & Toth, 1990; Winston & Goldberg, 2004). Masalah knapsack dapat diselesaikan dengan beberapa cara. Kita dapat menerapkan algoritma greedy untuk menyelesaikan masalah knapsack (Akçay et al., 2007). Furthermore, the knapsack problem can also be solved by an implicit enumeration algorithm. Some of the implicit enumeration algorithms are applicable to job-shop scheduling problems (Lageweg et al., 1977) and school desegregation problem (Liggett, 1973). In this paper we are interested to show how the implicit enumeration method solves the knapsack problem to form an optimal decision tree.

LITERATURE REVIEW

Binary Integer Progamming

Binary integer programming is a special case of integer programming, where each variable can only take on the value of 0 or 1. This may represent the selection or rejection of an option, the turning on or off of switches, a yes/no answer, or many other situations. In general, this binary integer programming problem can be modeled as follows:

Minimize
$$z = \sum_{j=1}^{n} c_j x_j$$

*Corresponding author

Subject to

$$\sum_{j=1}^{n} a_{ij}x_J \leq b_i\,, \quad i = 1,2,\dots,m$$

$$x_j = 0 \text{ or } 1, \;\; j = 1,2,\dots,n$$

Solving binary programming is actually easier to do by testing all the possibilities given that all the variables can only be 0 and 1. However, the test can take a long time if the number of variables is large enough. In general, if there are n variables, then there are 2^n possible solutions.

Decision Tree

A decision tree is a rooted tree T that consists of internal nodes representing attributes, leaf nodes representing labels, and edges representing the attributes' possible values. Decision tree classify instances by traversing from root node to leaf node. The classification process starts from root node of decision tree, tests the attribute specified by this node, and then moves down the tree branch according to the attribute value given (Cha & Tappert, 2008).

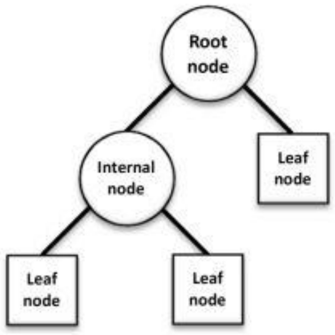


Figure 1. Decision tree structure

The concept of a decision tree is to convert data into a decision tree and decision rules. A decision tree is a set of if-then rules, where each path in the tree is associated with a rule where the premise consists of a set of nodes encountered and the conclusion of the rule consists of classes associated with the leaves of the path.

One of the applications of binary integer programming to form decision trees or decision making is the knapsack problem. The knapsack (backpack) problem arises when you have n items that can't all be put in one place (eg a bag/backpack). Each item has a different utility/benefit factor. The problem is choosing the items to carry (with limited space) so that the total utility is maximum.

METHOD

This research is a literature review. Given a knapsack problem and will be solved by implicit enumeration method. The basis of implicit enumeration is to enumerate a fraction of all possible solutions. Solutions that are not feasible or will produce a value that is not more optimal than the previously obtained value can be ignored. The implicit enumeration algorithm for solving binary integer programming (knapsack problem) is as follows (Morse et al., 2003; Sen, 2010):

1. **Forward Step**

   Test whether an iteration node needs to be branched. If necessary, branch the point to the left by taking a value of 1 to an independent variable. Keep doing it until you get to a point where you don't need to branch anymore.

2. **Backtracking**

   Look for the nearest node above it that has only a left branch (eg node v_k). If all the nodes above it already have 2 branches stop the process.

Branch the node v_k to the right by taking the value $x_k = 0$.
Return to step (1).

To reduce the number of iterations, at each node, a zero completion test and a feasibility test are carried out. Zero completion test is assigning a zero value to all independent variables. If the zero

*Corresponding author

solution is a feasible solution, the upper limit of the solution will be obtained. If it is not feasible, then a feasibility test must be carried out. The feasibility test is carried out by evaluating the maximum value of the slack variable on each constraint.If the maximum value of the slack variable in a constraint is negative ($s < 0$),, this means that it is not possible to obtain a feasible solution so that the node doesn't need to be branched. On the other hand, if the maximum value of s for all constraints is ≥ 0, it means that the point needs to be branched.

RESULT AND DISCUSSIONS

In this paper, we will use the knapsack problem with 3 variables. For example, given the knapsack problem as follows: An entrepreneur has 100 (million) funds to invest in construction, computer and foreign exchange businesses. The construction, computer and foreign exchange businesses require investments of 48, 39, and 25 (million) respectively and are expected to generate profits of 13, 10 and 5 (million). Some of the decision choices of this problem can be modeled as a binary integer programming model. The appropriate model for this problem is as follows:

Maximize $z = 13x_1 + 10x_2 + 5x_3$

Subject to $48x_1 + 39x_2 + 25x_3 \leq 100$ (1)

$x_1, x_2, x_3 = \{0,1\}$

where,

z = maximum profit

x_1 = construction business

x_2 = computer business

x_3 = foreign exchange business

and defined

$$x_i = \begin{cases} 1 & \text{if the } i^{\text{th}} \text{ investment is selected} \\ 0 & \text{if the } i^{\text{th}} \text{ investment is not selected} \end{cases}$$

$i = 1,2,3$

Because in this problem the model does not have a standard form of a binary integer programming, so it will be converted first into a standard form of a binary integer programming as follows:

1. Changing the objective function to minimize by multiplying the coefficient of the objective function by
(-1), we have

 Minimize $z = -13x_1 - 10x_2 - 5x_3$

 Subject to $48x_1 + 39x_2 + 25x_3 \leq 100$

 $x_1, x_2, x_3 = \{0,1\}$

2. Because the coefficient value in the objective function is negative, the transformation is carried out as follows:

$$x_1 = 1 - y_1$$
$$x_2 = 1 - y_2$$
$$x_3 = 1 - y_3$$

 Substituting into the objective and constraint functions, we have

 Minimize $z = -28 + 13y_1 + 10y_2 + 5y_3$

 Subject to $-48y_1 - 39y_2 - 25y_3 \leq -12$

- **Program 0**

 Minimize $z = -28 + 13y_1 + 10y_2 + 5y_3$

 Subject to $-48y_1 - 39y_2 - 25y_3 \leq -12$

 $y_1, y_2, y_3 = \{0,1\}$

Zero Completion Test

If $y_1 = y_2 = y_3 = 0$ then $\underline{z} = -28 < z_u = +\infty$ and on the constraint $0 \nleq -12$. Therefore, it is necessary to do a feasibility test.

*Corresponding author

Feasibility Test

$$s = -12 + 48y_1 + 39y_2 + 25y_3$$
$$s_{maks} = -12 + 48(1) + 39(1) + 25(1)$$
$$s_{maks} = 100 > 0$$

Since $s_{maks} > 0$ then node- 0 must be branched to the left by taking $y_1 = 1$ (Figure 2)

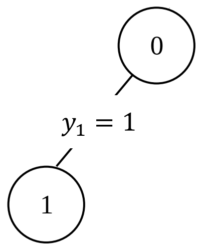


Figure 2. Branching to the left of node 0

- **Program 1**

Substituting the value of $y_1 = 1$ into program 0, we have:

Minimize $z = -15 + 10y_2 + 5y_3$
Subject to $-39y_2 - 25y_3 \leq 36$
$y_2, y_3 = \{0,1\}$

Zero Completion Test

If $y_2 = y_3 = 0$ then $\underline{z} = -15 < z_u = +\infty$ and on the constraint $0 \leq 36$. So the zero solution satisfies the constraint. This means that a new limit is obtained $z_u = \underline{z} = -15$ and backtracking is carried out by branching node 0 to the right and taking the value $y_1 = 0$ (Figure 3)

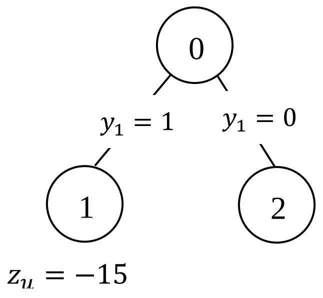


Figure 3. Branching to the right of node 0

- **Program 2**

Substituting the value of $y_1 = 0$ into program 0, we have:

Minimize $z = -28 + 10y_2 + 5y_3$
Subject to $-39y_2 - 25y_3 \leq -12$
$y_2, y_3 = \{0,1\}$

Zero Completion Test

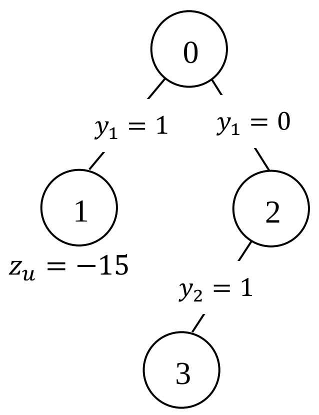
if $y_2 = y_3 = 0$, then $\underline{z} = -28 < z_u = -15$ and on the constraint $0 \nleq -12$. Therefore, it is necessary to do a feasibility test.

Feasibility Test

$$s = -12 + 39y_2 + 25y_3$$
$$s_{maks} = -12 + 39(1) + 25(1)$$
$$s_{maks} = 52 > 0$$

*Corresponding author

Since $s_{maks} > 0$, then node 2 must be branched to the left by taking $y_2 = 1$ (Figure 4).



Firgure 4. Branching to the left of node 2

- **Program 3**
  Substituting the value of $y_2 = 1$ into program 2, we have:
  Minimize $z = -18 + 5y_3$
  Subject to $-25y_3 \leq 27$
  $y_3 = \{0,1\}$

  Zero Completion Test
  If $y_3 = 0$, then $\underline{z} = -18 < z_u - 15$ and on the constraint $0 \leq 27$. So the zero solution satisfies the constraint. This means that a new limit is obtained $z_u = \underline{z} = -18$ and backtracking is carried out by branching node 2 to the right and taking the value $y_2 = 0$ (Figure 5).

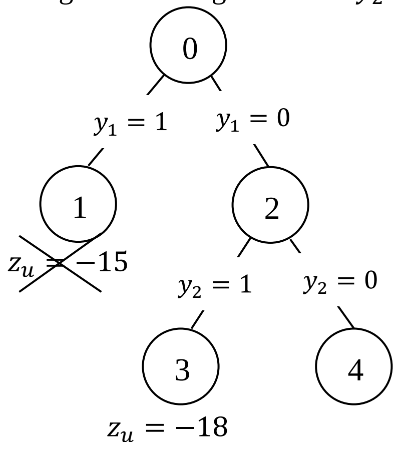


Figure 5. Branching to the right of node 2

- **Program 4**
  Substituting $y_2 = 0$ into program 2 , we have:
  Minimize $z = -28 + 5y_3$
  Subject to $-25y_3 \leq -12$
  $y_3 = \{0,1\}$

  Zero Completion Test
  If $y_3 = 0$ then $\underline{z} = -28 < z_u = -18$ and on the constraint$0 \nleq -12$. Therefore, it is necessary to do a feasibility test.
  Feasibility Test

$$s = -12 + 25y_3$$

*Corresponding author

$$s_{maks} = -12 + 25(1)$$
$$s_{maks} = 13 > 0$$

Since $s_{maks} \geq 0$ then node 4 2 must be branched to the left by taking $y_3 = 1$ (Figure 6).

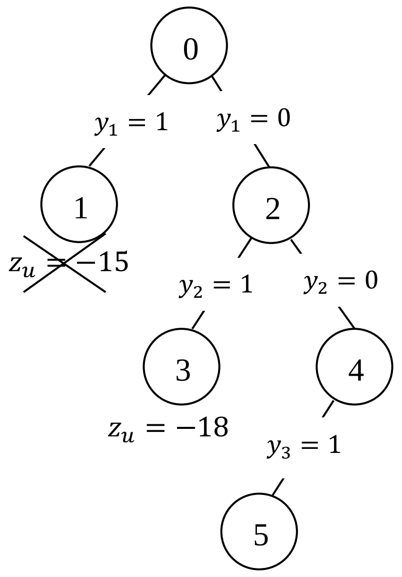


Figure 6. Branching to the left of node 4

- **Program 5**
  Substituting $y_3 = 1$ into program 4, we have:
  Minimize $z = -23$
  Subject to $-25 \leq -12$
  $y_3 = \{0,1\}$

  Zero Completion Test
  Since there are no more independent variables, then $\underline{z} = -23 < z_u = -18$ and the constraint is satisfied (true). So that the completion of zero is satisfied and a new upper limit is obtained, i.e., $z_u = \underline{z} = -23$ and an upward backtracking process is carried out by branching node 4 to the right which takes the value $y_3 = 0$ (Figure 6).

- **Program 6**
  Substituting $y_3 = 1$ into program 4, we have:
  Minimize $z = -28$
  Subject to $0 \leq -12$
  $y_3 = \{0,1\}$

  Zero Completion Test
  Since there are no more independent variables, then $\underline{z} = -28 < z_u = -23$ and the constraint is not satisfied, it is necessary to do a feasibility test.

  Feasibility Test
  $$s = -12$$
  $$s_{maks} = -12$$
  Since $s_{maks} < 0$, then node 6 doesn't need to be branched again and the backtracking process cannot be carried out because all nodes above node 6 already have 2 branches, which means the iteration process has been completed.

*Corresponding author

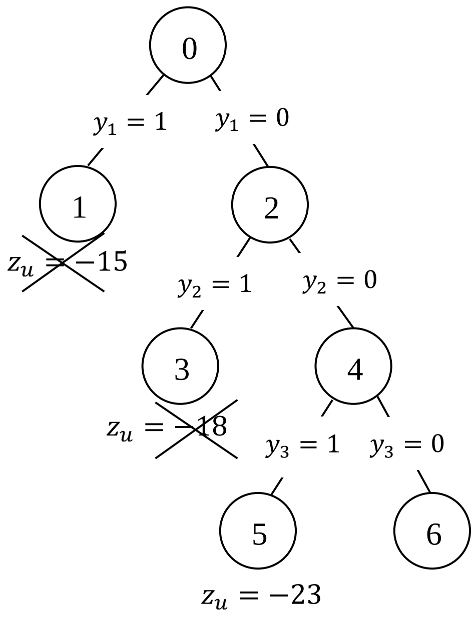


Figure 7. Branching to the right of node 4

Optimal Solution

The optimal solution is at node 5 with a value of $y_1 = 0, y_2 = 0, y_3 = 1$ and $z_u = -23$. If we back to the original problem then:

$$x_1 = 1 - y_1 = 1 - 0 = 1$$
$$x_2 = 1 - y_2 = 1 - 0 = 1$$
$$x_3 = 1 - y_3 = 1 - 1 = 0$$

with the maximum value $z_u = 23$.

So in order to obtain the maximum profit, the entrepreneur must invest (allocate) the funds to the construction business (x_1) of 48 million and the computer business (x_2) of 39 million, of which the maximum profit to be obtained is 23 million.

CONCLUSION

Trees can be used as interpretations in various disciplines. The use of trees as an analytical method in decision making is very useful and provides convenience in decision making. In the case of binary classification, decision making can be formulated in the form of a binary programming model and one of the effective methods for solving it is the implicit enumeration method, which has the advantage of simple coding and optimal solutions can be obtained without explicitly evaluating all possible solutions.

REFERENCES

Akçay, Y., Li, H. & Xu, S. H. (2007). Greedy algorithm for the general multidimensional knapsack problem. *Annals of Operations Research*, *150*(1), 17–29.

Cha, S.-H. & Tappert, C. C. (2008). Constructing Binary Decision Trees using Genetic Algorithms. *GEM*, 49–54.

Lageweg, B. J., Lenstra, J. K. & Rinnooy Kan, A. H. G. (1977). Job-shop scheduling by implicit enumeration. *Management Science*, *24*(4), 441–450.

Liggett, R. S. (1973). The application of an implicit enumeration algorithm to the school desegregation problem. *Management Science*, *20*(2), 159–168.

Martello, S. & Toth, P. (1990). *Knapsack problems: algorithms and computer implementations*. John Wiley & Sons, Inc.

Mitchell, T. M. & Mitchell, T. M. (1997). *Machine learning* (Vol. 1, Issue 9). McGraw-hill New York.

*Corresponding author

Morse, P. M., Kimball, G. E. & Gass, S. I. (2003). *Methods of operations research*. Courier Corporation.
Sen, R. P. (2010). Operations research: algorithms and applications. *PHI Learning*.
Winston, W. L. & Goldberg, J. B. (2004). *Operations research: applications and algorithms* (Vol. 3). Thomson Brooks/Cole Belmont.
Zhou, Z.-H. (2021). *Machine learning*. Springer Nature.

*Corresponding author