Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

# Uncertain bilevel knapsack problem based on an improved binary wolf pack algorithm*

Hu-sheng WU[1], Jun-jie XUE[2], Ren-bin XIAO[‡3], Jin-qiang HU[1]
[1]School of Equipment Management and Support, Armed Police Force Engineering University, Xi'an 710086, China
[2]Air Traffic Control and Navigation College, Air Force Engineering University, Xi'an 710051, China
[3]School of Artificial Intelligence and Automation, Huazhong University of Science and Technology, Wuhan 430074, China
E-mail: wuhusheng0421@163.com; 1019609875@qq.com; rbxiao@hust.edu.cn; hujinqiang002@163.com
Received Aug. 26, 2019; Revision accepted Dec. 2, 2019; Crosschecked Apr. 10, 2020; Published online June 9, 2020

**Abstract:** To address indeterminism in the bilevel knapsack problem, an uncertain bilevel knapsack problem (UBKP) model is proposed. Then, an uncertain solution for UBKP is proposed by defining the $\mathcal{P}_E$ Nash equilibrium and $\mathcal{P}_E$ Stackelberg–Nash equilibrium. To improve the computational efficiency of the uncertain solution, an evolutionary algorithm, the improved binary wolf pack algorithm, is constructed with one rule (wolf leader regulation), two operators (invert operator and move operator), and three intelligent behaviors (scouting behavior, intelligent hunting behavior, and upgrading). The UBKP model and the $\mathcal{P}_E$ uncertain solution are applied to an armament transportation problem as a case study.

**Key words:** Bilevel knapsack problem; Uncertainty; Improved binary wolf pack algorithm
https://doi.org/10.1631/FITEE.1900437 **CLC number:** TP18

## 1 Introduction

The bilevel knapsack problem (BKP) is a typical bilevel programming problem that models the hierarchical relationship between two autonomous, and possibly conflicting, decision-makers (the leader and the follower). BKP was first studied by Dempe and Richter (2000). In this problem, the follower solves a 0–1 knapsack problem subject to the capacity set by the leader. The leader earns a profit from the items selected by the follower, and both decision-makers seek to maximize their individual profits. BKP is formulated as a mixed-integer bilevel programming problem. Recently, researchers have increasingly focused on BKP, and several types of algorithms have been developed to solve BKP. Such algorithms include branch-and-cut (Özaltın et al., 2010), approximation (Qiu and Kern, 2015), dynamic programming (Brotcorne et al., 2013), heuristic optimization (Li et al., 2014), and polynomial (Carvalho et al., 2018). However, almost all previous algorithms were developed under deterministic assumptions.

BKP parameters are often indeterminate, particularly for issues related to combat, such as an armament transportation problem. To deal with problems under uncertain conditions, Liu B (2009b) developed the uncertainty theory, which provides the commonness of probability theory, credibility theory, and chance theory to handle uncertainty. When BKP parameters are uncertain variables, these types of BKPs are defined as uncertain bilevel knapsack problems (UBKPs).

‡ Corresponding author
* Project supported by the National Science and Technology Innovation 2030 Major Project of the Ministry of Science and Technology of China (No. 2018AAA0101200), the National Natural Science Foundation of China (No. 61502534), the Natural Science Foundation of Shaanxi Province, China (No. 2020JQ-493), and the Domain Foundation of China (No. 61400010304)
ORCID: Hu-sheng WU, https://orcid.org/0000-0003-0692-7467; Ren-bin XIAO, https://orcid.org/0000-0003-0951-2734

To handle UBKPs, an uncertain solution can be found by applying the $\mathcal{P}_E$ Nash equilibrium and $\mathcal{P}_E$ Stackelberg–Nash equilibrium to transform UBKP to a corresponding equivalent deterministic model. BKP is considered NP-hard (Caprara et al., 2014); therefore, a heuristic optimization algorithm should be applied to improve computational efficiency. A relatively new swarm intelligence based algorithm, the wolf pack algorithm (WPA), has been proposed and well developed (Wu et al., 2013; Li et al., 2016; Gao YJ et al., 2019). Due to its simplicity, effectiveness, and efficiency, WPA has been widely used (Xue et al., 2016; Bai et al., 2018; Han et al., 2018). In this study, the improved binary wolf pack algorithm (IBWPA) is designed to improve performance for UBKPs. IBWPA has one rule (wolf leader regulation), two operators (invert operator and move operator), and three intelligent behaviors (scouting behavior, intelligent hunting behavior, and upgrading).

## 2 Preliminaries

To deal with belief degree and uncertain problems rationally, several foundational definitions are introduced.

First, to depict the occurrence probability of event $\Lambda$, the uncertainty measurement is transformed to a real-valued space from 0 to 1, and a measurable real-value $\mathcal{M}\{\Lambda\}$ is determined for each event. To make real values $\mathcal{M}\{\Lambda\}$ mathematically significant, uncertain measurements must satisfy the limiting conditions to a certain degree.

**Axiom 1** (Normality axiom) (Cohn and Barnhart, 1998) Let $\Gamma$ be a nonempty set, and $\mathcal{M}\{\Gamma\}=1$ for the universal set $\Gamma$.

**Axiom 2** (Duality axiom) Let $\Lambda^{\mathrm{C}}$ denote that event $\Lambda$ does not occur, and $\mathcal{M}\{\Lambda\}+\mathcal{M}\{\Lambda^{\mathrm{C}}\}=1$ for any event $\Lambda$.

**Axiom 3** (Subadditivity axiom) For every countable sequence of events $\mathcal{L}\{\Lambda_1, \Lambda_2, \ldots\}$, $\mathcal{M}\{\Lambda\}$ satisfies the following formula:

$$\mathcal{M}\left\{\sum_{i=1}^{\infty}\Lambda_i\right\}\leq\sum_{i=1}^{\infty}\mathcal{M}\{\Lambda_i\}. \tag{1}$$

**Axiom 4** (Product axiom) (Liu B, 2009a) The triplet $(\Gamma_k, \mathcal{L}_k, \mathcal{M}_k)$ ($k$=1, 2, …) is called an uncertainty space. Let $\Gamma=\Gamma_1\times\Gamma_2\times\ldots$ be the set of ordered tuples composed of nonempty sets. Let $\Lambda=\Lambda_1\times\Lambda_2\times\ldots$ be the set of events in the set $\Gamma$ of ordered tuples, and $\mathcal{L}=\mathcal{L}_1\times\mathcal{L}_2\times\ldots$ be the product $\sigma$-algebra of set $\Gamma$. Then, the product uncertainty measure $\mathcal{M}$ is an uncertain measure on the product $\sigma$-algebra $\mathcal{L}$ that satisfies the following:

$$\mathcal{M}\left\{\prod_{i=1}^{\infty}\Lambda_i\right\}=\bigwedge_{i=1}^{\infty}\mathcal{M}\{\Lambda_i\}. \tag{2}$$

The set function $\mathcal{M}$ is called the product uncertainty measure of uncertain spaces $(\Gamma_k, \mathcal{L}_k, \mathcal{M}_k)$ ($k$=1, 2, …). The product uncertainty measure accords with the three basic axioms (Axioms 1–3) of the uncertainty measure and is a generalized uncertainty measure to analyze uncertainty problems in multiple uncertain spaces.

Second, the uncertain variable is a measurable function in the uncertain space. Its exact mathematical definition comprises the following definitions.

**Definition 1** (Uncertainty measures) Let $\Gamma$ be a nonempty set, the $\sigma$-algebra of event $\Lambda$ in set $\Gamma$ the elements of $\mathcal{L}$, and $\mathcal{M}$ the set function from the real-value $\mathcal{L}$ to the [0, 1] space. When the set function $\mathcal{M}$ satisfies Axioms 1–3, the real-value set function $\mathcal{M}\{\cdot\}$ is called the uncertainty measurement.

**Definition 2** (Uncertainty variables) Let $\xi$ be a function that maps from an uncertain space $(\Gamma, \mathcal{L}, \mathcal{M})$ to a set of real numbers. For any $B$ that is a Borel set, the set $\xi$ ($\xi\in B$) is an event in $\mathcal{L}$, and then $\xi$ is an uncertain variable.

One key factor that affects the system performance is whether or not the parameters are independent. Therefore, it is necessary to study the independence of uncertain parameters in the uncertain space.

**Definition 3** (Uncertain variable independence) Given that $\xi_1, \xi_2, \ldots, \xi_m$ are uncertain variables in an uncertain space $(\Gamma, \mathcal{L}, \mathcal{M})$, for any set $B_i$ ($i$=1, 2, …, $m$)

in Borel, if the uncertain variables satisfy

$$\mathcal{M}\left\{\bigcap_{i=1}^{m}\left(\xi_i \in B_i\right)\right\} = \min_{i=1,2,\cdots,m} \mathcal{M}\left(\xi_i \in B_i\right), \tag{3}$$

then the uncertain variables are independent of each other.

**Definition 4** (Uncertainty distribution) (Liu B, 2016) The uncertainty distribution $\Phi$ of an uncertain variable $\xi$ is defined as

$$\Phi(x) = \mathcal{M}\{\xi \le x\} \tag{4}$$

for any real number $x$.

Two basic forms of an uncertainty distribution are described in the following.

1. The linear uncertainty distribution $\mathcal{L}(a, b)$ is defined as

$$\Phi(x) = \begin{cases} 0, & x \le a, \\ \dfrac{x-a}{b-a}, & a < x \le b, \\ 1, & x > b, \end{cases} \tag{5}$$

where $a$ and $b$ are real numbers satisfying $a<b$.

2. The zigzag uncertainty distribution $\mathcal{Z}(a, b, c)$ is defined by

$$\Phi(x) = \begin{cases} 0, & x \le a, \\ \dfrac{x-a}{2(b-a)}, & a < x \le b, \\ \dfrac{x+c-2b}{2(c-b)}, & b < x \le c, \\ 1, & x > c, \end{cases} \tag{6}$$

where $a$, $b$, and $c$ are real numbers satisfying $a<b<c$.

**Definition 5** (Inverse uncertainty distribution) (Liu B, 2016) Let $\xi$ be an uncertain variable with regular uncertainty distribution $\Phi$. Then, the inverse function $\Phi^{-1}$ is called the inverse uncertainty distribution of $\xi$.

**Theorem 1** (Liu B, 2010) Let $\xi$ be an uncertain variable with regular uncertainty distribution $\Phi$. Then, the expected value of uncertain variable $\xi$ is

$$E(\xi) = \int_0^1 \Phi^{-1}(\alpha)\mathrm{d}\alpha. \tag{7}$$

**Theorem 2** (Liu B, 2010) Let $\xi_1, \xi_2, \ldots, \xi_n$ be independent uncertain variables with regular uncertainty distributions $\Phi_1, \Phi_2, \ldots, \Phi_n$, respectively. If the function $f(x_1, x_2, \ldots, x_n)$ is strictly increasing with respect to $x_1, x_2, \ldots, x_m$ and strictly decreasing with respect to $x_{m+1}, x_{m+2}, \ldots, x_n$, then $\xi=f(x_1, x_2, \ldots, x_n)$ has an inverse uncertainty distribution:

$$\begin{aligned} \Psi^{-1}(\alpha) = f\Big(&\Phi_1^{-1}(\alpha), \Phi_2^{-1}(\alpha), \ldots, \Phi_m^{-1}(\alpha), \\ &\Phi_{m+1}^{-1}(1-\alpha), \Phi_{m+2}^{-1}(1-\alpha), \ldots, \Phi_n^{-1}(1-\alpha)\Big). \end{aligned} \tag{8}$$

**Definition 6** (Expected-value principle $\mathcal{P}_E$) (Wang ZT et al., 2015) Let $\xi$ and $\eta$ be two uncertain variables. Then, we say $\xi \preceq (\text{or} \prec) \eta$ if and only if $E[\xi] \le$ (or $<$) $E[\eta]$, where $E[\cdot]$ denotes the expected value of the uncertain variable.

Third, to solve a UBKP effectively without changing the uncertain environment, the concept of constraints and the optimal solution of UBKP must be defined. An uncertain solution for a UBKP is proposed by defining the uncertain-constraint principle, $\mathcal{P}_E$ Nash equilibrium, and $\mathcal{P}_E$ Stackelberg–Nash equilibrium.

**Definition 7** (Uncertain-constraint principle) $g^1(\xi^1) \preceq \eta^1$ is the uncertainty constraint in the top-level decision-making system of BKP, and $g_i^2(\xi_i^2) \preceq \eta_i^2$ is the uncertainty constraint in the $i^{\text{th}}$ bottom-level decision-making system. $\alpha^1$ is defined as the degree of confidence factor of the top-level decision-making system of BKP, and $\alpha_i^2$ is the degree of confidence factor of the $i^{\text{th}}$ bottom-level decision-making system. The uncertain chance constraints of the top-level decision-making system and the $i^{\text{th}}$ bottom-level decision-making system are represented as

$$\mathcal{M}\left\{g^1\left(\xi^1\right) \preceq \eta^1\right\} \ge \alpha^1, \tag{9}$$

$$\mathcal{M}\left\{g^1\left(\xi^1\right) \preceq \eta^1 \cap g_i^2\left(\xi_i^2\right) \preceq \eta_i^2\right\} \ge \alpha_i^2, \tag{10}$$

where degrees of confidences $\alpha^1$ and $\alpha_i^2$ took the same value (95%).

If the constraints $g^1(\xi^1) \preceq \eta^1$ and $g_i^2(\xi_i^2) \preceq \eta_i^2$ are independent of each other, then inequality (10) can be transformed as

$$\min\left(\mathcal{M}\left\{g^1\left(\xi^1\right) \preceq \eta^1\right\}, \mathcal{M}\left\{g_i^2\left(\xi_i^2\right) \preceq \eta_i^2\right\}\right) \geq \alpha_i^2. \quad (11)$$

**Definition 8** ($\mathcal{P}_E$ Nash equilibrium) (Wang ZT et al., 2015) Let $\boldsymbol{x}$ be a feasible decision vector of the leader. A $\mathcal{P}_E$ Nash equilibrium of followers is defined as the feasible array $\left(\boldsymbol{y}_1^*, \boldsymbol{y}_2^*, \cdots, \boldsymbol{y}_k^*\right)$ with respect to $\boldsymbol{x}$ if there is no feasible array $\left(\boldsymbol{y}_1^*, \boldsymbol{y}_2^*, ..., \boldsymbol{y}_{i-1}^*, \boldsymbol{y}^*, \boldsymbol{y}_{i+1}^*, ..., \boldsymbol{y}_k^*\right)$ such that

$$\begin{aligned} &E\left[f_i\left(\boldsymbol{x}, \boldsymbol{y}_1^*, \boldsymbol{y}_2^*, ..., \boldsymbol{y}_{i-1}^*, \boldsymbol{y}_i, \boldsymbol{y}_{i+1}^*, ..., \boldsymbol{y}_k^*, \xi\right)\right] \\ &\geq E\left[f_i\left(\boldsymbol{x}, \boldsymbol{y}_1^*, \boldsymbol{y}_2^*, ..., \boldsymbol{y}_{i-1}^*, \boldsymbol{y}_i^*, \boldsymbol{y}_{i+1}^*, ..., \boldsymbol{y}_k^*, \xi\right)\right]. \end{aligned} \quad (12)$$

**Definition 9** ($\mathcal{P}_E$ Stackelberg–Nash equilibrium) Let $\boldsymbol{x}^*$ be a feasible decision vector of the leader and $\left(\boldsymbol{y}_1^*, \boldsymbol{y}_2^*, ..., \boldsymbol{y}_k^*\right)$ the $\mathcal{P}_E$ Nash equilibrium of followers with respect to $\boldsymbol{x}^*$. A $\mathcal{P}_E$ Stackelberg–Nash equilibrium for the bilevel problem is defined as $\left(\boldsymbol{x}^*, \boldsymbol{y}_1^*, \boldsymbol{y}_2^*, ..., \boldsymbol{y}_k^*\right)$ if there is no feasible array $\left(\boldsymbol{x}^{**}, \boldsymbol{y}_1^{**}, \boldsymbol{y}_2^{**}, ..., \boldsymbol{y}_k^{**}\right)$ such that

$$\begin{aligned} &E\left[F\left(\boldsymbol{x}^{**}, \boldsymbol{y}_1^{**}, \boldsymbol{y}_2^{**}, ..., \boldsymbol{y}_k^{**}, \xi\right)\right] \\ &\geq E\left[F\left(\boldsymbol{x}^*, \boldsymbol{y}_1^*, \boldsymbol{y}_2^*, ..., \boldsymbol{y}_k^*, \xi\right)\right], \end{aligned} \quad (13)$$

where $\left(\boldsymbol{y}_1^{**}, \boldsymbol{y}_2^{**}, ..., \boldsymbol{y}_k^{**}\right)$ is the $\mathcal{P}_E$ Nash equilibrium of followers with respect to $\boldsymbol{x}^{**}$.

## 3 Problem formulation

In this section, UBKP is described. Note that all related parameters are uncertain variables. First, mathematical descriptions of BKP and UBKP are proposed. Then, an equivalent deterministic model of UBKP is obtained.

### 3.1 Formulation of BKP

To describe UBKP clearly, BKP is introduced first. BKP involves two decision-makers interacting hierarchically. A leader (responsible for a subset of variables) must explicitly consider the reaction of a follower (responsible for another subset of variables) in its optimization process. The specificity of BKP is that the follower problem is a knapsack problem and the leader's constraints involve both leader and follower variables. In other words, BKP is an optimization problem whose feasible set is determined by the set of optimal solutions of a parametric knapsack problem (KP). A BKP can be appropriately modeled as the following linear bilevel problem with integer variables:

$$\begin{aligned} &\max_{(\boldsymbol{x},\boldsymbol{y})} F(\boldsymbol{x}, \boldsymbol{y}, \xi) = \boldsymbol{d}^1\boldsymbol{x} + \boldsymbol{d}^2\boldsymbol{y} \\ &\text{s.t.} \begin{cases} \boldsymbol{B}^1\boldsymbol{x} + \boldsymbol{B}^2\boldsymbol{y} \leq c^1, \\ \boldsymbol{y} = (\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_k), \end{cases} \end{aligned} \quad (14)$$

where $\boldsymbol{y}_i$ solves

$$\max_{\boldsymbol{y}_i} f_i\left(\boldsymbol{y}_i'\right) = \boldsymbol{d}_i^3\boldsymbol{y}_i' \ \text{ s.t. } \ \boldsymbol{b}_i^1\boldsymbol{x} + \boldsymbol{b}_i^2\boldsymbol{y}_i' \leq c_i^2.$$

In problem (14), $\boldsymbol{x} \in \{0,1\}^{N^1}$ is the decision parameter vector of the top-level decision-maker, where $N^1$ is the number of parameters that the top-level decision-maker requires to make decisions. $\boldsymbol{y}$=($\boldsymbol{y}_1$, $\boldsymbol{y}_2$, …, $\boldsymbol{y}_k$) is the decision vector composed of all the decision parameter variables of all the bottom decision-makers, and the bottom decision-maker $i$ is the decision parameter vector $\boldsymbol{y}_i \in \{0,1\}^{N_i^2}$ where $N_i^2$ is the number of parameters that the bottom decision-maker $i$ needs to determine. $\boldsymbol{d}^1$ is the weight vector of the contribution of the top decision-maker's decision parameter vector $\boldsymbol{x}$ to the value of the whole system, $\boldsymbol{d}^2$ is the weight vector of the contribution of the bottom decision-maker's decision parameter vector $\boldsymbol{y}$ to the value of the whole system, $\boldsymbol{B}^1$ is the resource consumption coefficient (system-level constraint coefficient) of the top-level decision vector $\boldsymbol{x}$ for the whole system, $\boldsymbol{B}^2$ is the resource consumption coefficient of all the underlying decision vectors $\boldsymbol{y}$ for the whole system, $c^1$ is the set of total resource constraints of the whole system, $\boldsymbol{d}_i^3$ is the weight vector of the contribution of the top-level decision parameter vector $\boldsymbol{y}_i'$ to the value of the subsystem in the bottom decision-making vector $i$, $\boldsymbol{b}_i^1$ is the resource consumption coefficient of the top decision-making parameter vector $\boldsymbol{x}$ in the bottom decision-making subsystem $i$, $\boldsymbol{b}_i^2$ is the resource consumption coefficient of vector pair $\boldsymbol{y}_i'$ of the bottom decision-making parameter in the bottom decision-making subsystem $i$, and $c_i^2$ is the set of resource constraints of the bottom decision subsystem.

To facilitate analysis and calculation, the top layer constraint condition and the parameter part of the $i^{\text{th}}$ bottom layer constraint condition in BKP can be respectively expressed as follows:

$$G(\boldsymbol{x},\boldsymbol{y}) = \boldsymbol{B}^1\boldsymbol{x} + \boldsymbol{B}^2\boldsymbol{y} - c^1, \tag{15}$$

$$g_i\left(\boldsymbol{y}_i'\right) = \boldsymbol{b}_i^1\boldsymbol{x} + \boldsymbol{b}_i^2\boldsymbol{y}_i' - c_i^2. \tag{16}$$

Obviously, when the constraints of the top or bottom decision-maker in BKP are slack ($c^1=\infty$ or $c_i^2=\infty$), BKP is transformed to an ordinary KP. Therefore, BKP exhibits the same NP-hard characteristics as the ordinary KP.

### 3.2 Formulation of UBKP

UBKP is special in BKP. The related parameters involved in the ordinary BKP are all determined. In contrast, parameters in the uncertain double-layer knapsack problem are uncertain. Even some system environment parameters are uncertain, starting from the mathematical model of the ordinary BKP, the mathematical model of UBKP can be obtained by extending the relevant deterministic parameters into the uncertain parameters.

When the involved parameters are uncertain, UBKP can be expressed as follows:

$$\begin{aligned} &\max_{(\boldsymbol{x},\boldsymbol{y})} F(\boldsymbol{x},\boldsymbol{y},\xi) = \boldsymbol{\delta}^1\boldsymbol{x} + \boldsymbol{\delta}^2\boldsymbol{y} \\ &\text{s.t.}\ \begin{cases} G = \left\{\mathcal{B}^1\boldsymbol{x} + \mathcal{B}^2\boldsymbol{y} - \chi^1 \preceq 0\right\}, \\ \boldsymbol{y} = (\boldsymbol{y}_1,\boldsymbol{y}_2,\dots,\boldsymbol{y}_k), \end{cases} \end{aligned} \tag{17}$$

where $\boldsymbol{y}_i$ solves

$$\begin{aligned} &\max_{\boldsymbol{y}_i'} f_i\left(\boldsymbol{y}_i',\xi\right) = \boldsymbol{\delta}_i^3\boldsymbol{y}_i' \\ &\text{s.t.}\ g_i = \left\{\boldsymbol{\beta}_i^1\boldsymbol{x} + \boldsymbol{\beta}_i^2\boldsymbol{y}_i' - \chi_i^2 \preceq 0\right\}, \end{aligned}$$

and $\xi=\{\boldsymbol{\delta}^1, \boldsymbol{\delta}^2, \mathcal{B}^1, \mathcal{B}^2, \chi^1, \boldsymbol{\delta}_i^3, \boldsymbol{\beta}_i^1, \boldsymbol{\beta}_i^2, \chi_i^2\}$ is the set of uncertain parameters corresponding to parameters $\boldsymbol{d}^1$, $\boldsymbol{d}^2$, $\boldsymbol{B}^1$, $\boldsymbol{B}^2$, $c^1$, $\boldsymbol{d}_i^3$, $\boldsymbol{b}_i^1$, $\boldsymbol{b}_i^2$, and $c_i^2$ respectively in the BKP model.

Simultaneously, to facilitate analysis and calculation, the top layer and $i^{\text{th}}$ bottom layer constraint conditions in UBKP are expressed as given in Eqs. (15) and (16), respectively.

Due to these constraints, the objective function of BKP has some infeasible regions. The objective function can be simplified effectively using the penalty function method, which transforms the constraints into penalty factors. Then, the mathematical model of UBKP can be transformed as follows:

$$\begin{aligned} &\max_{(\boldsymbol{x},\boldsymbol{y})} F(\boldsymbol{x},\boldsymbol{y},\xi) = \rho^1(\boldsymbol{\delta}^1\boldsymbol{x} + \boldsymbol{\delta}^2\boldsymbol{y}) \\ &\text{s.t.}\ \boldsymbol{y} = (\boldsymbol{y}_1,\boldsymbol{y}_2,\dots,\boldsymbol{y}_k), \end{aligned} \tag{18}$$

where $\boldsymbol{y}_i$ solves $\max_{\boldsymbol{y}_i'} f_i\left(\boldsymbol{y}_i',\xi\right) = \rho_i^2\left(\boldsymbol{\delta}_i^3\boldsymbol{y}_i'\right)$. Here, $\rho^1$ is the top penalty factor derived from the transformation of constraints in top-level decision systems and $\rho_i^2$ is the $i^{\text{th}}$ bottom penalty factor obtained from the transformation of the constraints of the $i^{\text{th}}$ bottom-level decision system, expressed as follows:

$$\rho^1 = \begin{cases} 1, & G(\boldsymbol{x},\boldsymbol{y},\xi) \preceq 0, \\ 0, & \text{otherwise}, \end{cases} \tag{19}$$

$$\rho_i^2 = \begin{cases} 1, & g_i\left(\boldsymbol{y}_i',\xi\right) \preceq 0, \\ 0, & \text{otherwise}. \end{cases} \tag{20}$$

### 3.3 Equivalent deterministic model

The objective functions are uncertain variables, and the constraint conditions are uncertain inequality constraints; therefore, the problem cannot be maximized directly. According to the $\mathcal{P}_E$ Nash equilibrium and $\mathcal{P}_E$ Stackelberg–Nash equilibrium, UBKP can be solved with the Stackelberg–Nash equilibrium solution rather than an uncertain solution. Then, the UBKP model can be transferred to its expected model; i.e., its expected value can be maximized subject to a set of chance constraints:

$$\begin{aligned}&\max_{(x,y)} E[F(\boldsymbol{x},\boldsymbol{y},\xi)]=\rho^1\cdot E\left[\boldsymbol{\delta}^1\boldsymbol{x}+\boldsymbol{\delta}^2\boldsymbol{y}\right]\\&\quad\text{s.t. } \boldsymbol{y}=(\boldsymbol{y}_1,\boldsymbol{y}_2,\ldots,\boldsymbol{y}_k),\end{aligned}\tag{21}$$

where $\boldsymbol{y}_i$ solves $\max\limits_{\boldsymbol{y}_i'} E\left[f_i\left(\boldsymbol{y}_i',\xi\right)\right]=\rho_i^2 E\left[\boldsymbol{\delta}_i^3\boldsymbol{y}_i'\right]$, and $\rho^1$ and $\rho_i^2$ are penalty factors expressed as follows:

$$\rho^1=\begin{cases}1, & \mathcal{M}\{G(\boldsymbol{x},\boldsymbol{y},\xi)\preceq 0\}\geq\alpha^1,\\0, & \text{otherwise},\end{cases}\tag{22}$$

$$\rho_i^2=\begin{cases}1, & \mathcal{M}\{G(\boldsymbol{x},\boldsymbol{y},\xi)\preceq 0\cap g_i(\boldsymbol{y}_i',\xi)\preceq 0\}\geq\alpha_i^2,\\0, & \text{otherwise}.\end{cases}\tag{23}$$

From a mathematical perspective, the expected UBKP model cannot be solved directly. It is essential to transform it to a corresponding equivalent deterministic BKP model.

Let $\xi_1, \xi_2, \ldots, \xi_m$ be independent uncertain variables with regular uncertainty distributions $\varPhi_1, \varPhi_2, \ldots, \varPhi_m$, respectively. The real function $F$ is the objective function of the top decision system in BKP. We assume that leader function $F$ strictly increases relative to $\xi_1, \xi_2, \ldots, \xi_j$ and strictly decreases relative to the following:

$$\begin{aligned}&E\left[F\left(\boldsymbol{x},\boldsymbol{y},\xi_1,\xi_2,\ldots,\xi_j,\xi_{j+1},\ldots,\xi_m\right)\right]\\&=\rho^1\int_0^1 F\left(\boldsymbol{x},\boldsymbol{y},\varPhi_1^{-1}(\alpha),\varPhi_2^{-1}(\alpha),\ldots,\right.\\&\quad\left.\varPhi_j^{-1}(\alpha),\varPhi_{j+1}^{-1}(1-\alpha),\ldots,\varPhi_m^{-1}(1-\alpha)\right)\mathrm{d}\alpha.\end{aligned}\tag{24}$$

Leader constraint function $G$ strictly increases relative to $\xi_1, \xi_2, \ldots, \xi_l$ and strictly decreases relative to $\xi_{l+1}, \xi_{l+2}, \ldots, \xi_m$. Then, the uncertainty constraint $\mathcal{M}\left\{g^1(\xi^1)\preceq\eta^1\right\}\geq\alpha^1$ can be converted into the following expected form:

$$\begin{aligned}&\int_0^1 G\left(\boldsymbol{x},\boldsymbol{y},\varPhi_1^{-1}(\alpha),\varPhi_2^{-1}(\alpha),\ldots,\varPhi_l^{-1}(\alpha),\right.\\&\quad\left.\varPhi_{l+1}^{-1}(1-\alpha),\ldots,\varPhi_m^{-1}(1-\alpha)\right)\mathrm{d}\alpha\leq 0.\end{aligned}\tag{25}$$

The penalty function of the uncertain constraints of the top-level decision system can be obtained as follows:

$$\rho^1=\begin{cases}1, & \int_0^1 G\left(\boldsymbol{x},\boldsymbol{y},\varPhi_1^{-1}(\alpha),\varPhi_2^{-1}(\alpha),\ldots,\varPhi_l^{-1}(\alpha),\right.\\ & \quad\left.\varPhi_{l+1}^{-1}(1-\alpha),\ldots,\varPhi_m^{-1}(1-\alpha)\right)\mathrm{d}\alpha\leq 0,\\0, & \text{otherwise}.\end{cases}\tag{26}$$

Similarly, for the underlying decision-making system in UBKP, the objective function needs to be transformed to the corresponding equivalent deterministic objective function. Let the real function $f_i$ be the objective function of the bottom-level decision-making system in BKP. Without loss of generality, $f_i$ strictly increases relative to $\xi_1, \xi_2, \ldots, \xi_{j_i}$ and strictly decreases relative to $\xi_{j_i+1},\xi_{j_i+2},\ldots,\xi_m$. Then, $f_i$ can be transformed as follows:

$$\begin{aligned}&E\left[f_i\left(\boldsymbol{x},\boldsymbol{y},\xi_1,\xi_2,\ldots,\xi_{j_i},\xi_{j_i+1},\ldots,\xi_m\right)\right]\\&=\rho_i^2\int_0^1 F\left(\boldsymbol{x},\boldsymbol{y},\varPhi_1^{-1}(\alpha),\varPhi_2^{-1}(\alpha),\ldots,\right.\\&\quad\left.\varPhi_{j_i}^{-1}(\alpha),\varPhi_{j_i+1}^{-1}(1-\alpha),\ldots,\varPhi_m^{-1}(1-\alpha)\right)\mathrm{d}\alpha.\end{aligned}\tag{27}$$

Constraint function $g_i$ strictly increases relative to $\xi_1, \xi_2, \ldots, \xi_{c_i}$ and strictly decreases relative to $\xi_{c_i+1},\xi_{c_i+2},\ldots,\xi_m$. Then, the uncertainty constraint $\mathcal{M}\left\{g_i^2(\boldsymbol{x},\boldsymbol{y},\xi)\preceq 0\right\}\geq\alpha_i^2$ can be converted as follows:

$$\begin{aligned}&\int_0^1 g_i^2\left(\boldsymbol{x},\boldsymbol{y},\varPhi_1^{-1}(\alpha),\varPhi_2^{-1}(\alpha),\ldots,\varPhi_{c_i}^{-1}(\alpha),\right.\\&\quad\left.\varPhi_{c_i+1}^{-1}(1-\alpha),\ldots,\varPhi_m^{-1}(1-\alpha)\right)\mathrm{d}\alpha\leq 0.\end{aligned}\tag{28}$$

The penalty function of the uncertain constraints of the bottom-level decision system can be obtained as follows:

$$\rho_i^2=\begin{cases}1, & \int_0^1 g_i^2\left(\boldsymbol{x},\boldsymbol{y},\varPhi_1^{-1}(\alpha),\varPhi_2^{-1}(\alpha),\ldots,\right.\\ & \quad\left.\varPhi_{c_i}^{-1}(\alpha),\varPhi_{c_i+1}^{-1}(1-\alpha),\ldots,\varPhi_m^{-1}(1-\alpha)\right)\mathrm{d}\alpha\leq 0,\\0, & \text{otherwise}.\end{cases}\tag{29}$$

According to Theorems 1 and 2, the UBKP model in Eq. (17) can be transformed to an equivalent deterministic BKP model by integrating Eqs. (24), (26), (27), and (29). The UBKP model is expressed as follows:

$$\max_{(\boldsymbol{x},\boldsymbol{y})}\Big(\rho^1\int_0^1 F\big(\boldsymbol{x},\boldsymbol{y},\Phi_1^{-1}(\alpha),\Phi_2^{-1}(\alpha),...,\Phi_j^{-1}(\alpha),\Phi_{j+1}^{-1}(1-\alpha),...,\Phi_m^{-1}(1-\alpha)\big)\mathrm{d}\alpha\Big)$$

$$\text{s.t.}\ \begin{cases}\rho^1=\begin{cases}1, & \int_0^1 G\big(\boldsymbol{x},\boldsymbol{y},\Phi_1^{-1}(\alpha),\Phi_2^{-1}(\alpha),...,\Phi_{j_G}^{-1}(\alpha),\Phi_{j_G+1}^{-1}(1-\alpha),...,\Phi_m^{-1}(1-\alpha)\big)\mathrm{d}\alpha\le 0,\\ 0, & \text{otherwise},\end{cases}\\ \boldsymbol{y}=(\boldsymbol{y}_1,\boldsymbol{y}_2,...,\boldsymbol{y}_k),\end{cases} \tag{30}$$

where $\boldsymbol{y}_i$ solves

$$\max_{(\boldsymbol{x},\boldsymbol{y})}\Big(\int_0^1 \rho_i^2 f_i\big(\boldsymbol{x},\boldsymbol{y},\Phi_1^{-1}(\alpha),\Phi_2^{-1}(\alpha),...,\Phi_{j_i}^{-1}(\alpha),\Phi_{j_i+1}^{-1}(1-\alpha),...,\Phi_m^{-1}(1-\alpha)\big)\mathrm{d}\alpha\Big)$$

$$\text{s.t.}\ \rho^1=\begin{cases}1, & \Big(\int_0^1 g_i\big(\boldsymbol{x},\boldsymbol{y},\Phi_1^{-1}(\alpha),\Phi_2^{-1}(\alpha),...,\Phi_{c_i}^{-1}(\alpha),\Phi_{c_i+1}^{-1}(1-\alpha),...,\Phi_m^{-1}(1-\alpha)\big)\mathrm{d}\alpha\le 0\Big)\cap\big(\rho^1=1\big),\\ 0, & \text{otherwise}.\end{cases}$$

## 4 Improved binary wolf pack algorithm

In this section, the evolutionary IBWPA is proposed, comprising one regulation, two operators, and three intelligent behaviors.

### 4.1 Description of IBWPA

The binary smart wolf pack space is an $n\times m$ Euclidean space, where $n$ is the population of artificial wolves and $m$ the binary coding length of an artificial wolf. The location of an element (artificial wolf) $i$ is $\boldsymbol{I}_i(x)=(x_{i,1}, x_{i,2}, \ldots, x_{i,m})$, and the resource abundance (objective function value) of an element (artificial wolf) $i$ is $Y_i=f(\boldsymbol{I}_i(x))$, where $x_{i,j}$ represents the number of copies of item $j$. IBWPA comprises one regulation (wolf leader regulation), two operators (invert operator and move operator), and three intelligent behaviors (scouting, intelligent hunting, and upgrading). Initially, the first-generation location of an artificial wolf is generated randomly.

#### 4.1.1 Wolf leader regulation

The wolf leader is a unique wolf that has overall optimal resources and commands other artificial wolves to cooperate efficiently. To ensure effectiveness of the command, there should be only a single wolf leader. If more than one wolf has the same optimal resources, one is randomly selected as the wolf leader, and the others are regenerated based on the upgrading behavior. In addition, to ensure command efficiency, the wolf leader does not participate in the intelligent hunting behavior.

#### 4.1.2 Invert operator

For each item, we denote its copy number as binary code 1 when this item is in the knapsack; otherwise, denote it as binary code 0. To invert the characters of binary code 0 to 1 (or 1 to 0) effectively (i.e., to change the status of item $j$, whether or not it is in the knapsack), we propose the inverse operator $\aleph(\cdot)$:

$$\aleph(\cdot)=\begin{cases}0, & x_{i,j}=1,\\ 1, & x_{i,j}=0,\end{cases} \tag{31}$$

where $x_{i,j}$ is the $j^{\text{th}}$ binary character (or the number of copies of item $j$) of artificial wolf $i$.

#### 4.1.3 Move operator

The move operator describes the movement of an artificial wolf. Here, let $\boldsymbol{I}_i(x)=(x_{i,1}, x_{i,2}, \ldots, x_{i,m})$ be the position of an artificial wolf, $M$ the moving space of an artificial wolf (i.e., a set of binary characters that may be inverted), and $r$ the step of an artificial wolf (i.e., the number of binary characters to be inverted). $\Theta(\boldsymbol{I}_i, M, r)$ means that $r$ binary characters randomly selected from set $M$ are inverted by the invert operator. As shown in Fig. 1, $\boldsymbol{I}_i$=(1, 1, 1, 0, 0, 1), $M$={3, 5, 6}, and $r$=2. Then, $\Theta(\boldsymbol{I}_i, M, r)$ is one of items (1, 1, 0, 0, 1, 1), (1, 1, 0, 0, 0, 0), and (1, 1, 1, 0, 1, 0).

#### 4.1.4 Scouting behavior

Artificial wolf $W_i$ shows great sensitivity to its ambient, and thus we propose the scouting behavior in each generation of the foraging process. Here, artificial wolf $W_i$ observes resources of $h$ different positions (items) to obtain a better resource position. The $p^{\text{th}}$ item of artificial wolf $W_i$ is defined as follows:

$$\boldsymbol{I}_i^p = \Theta\left(\boldsymbol{I}_i, M, \text{Step}_a\right), \tag{32}$$

where $M$={1, 2, …, $m$} is the set with all serial numbers of the binary characters and $\text{Step}_a$ the scouting step coefficient.

We then obtain $h$ different positions (items) $\boldsymbol{I}_i=\left\{\boldsymbol{I}_i^1, \boldsymbol{I}_i^2, ..., \boldsymbol{I}_i^h\right\}$ and resources (objective value) of the $h$ positions (items) $Y_i=\left\{Y_i^1, Y_i^2, ..., Y_i^h\right\}$. Artificial wolf $W_i$ moves to the position (item) with the maximum resource (objective value), i.e., the position with the resource (objective value) greater (larger) than that artificial wolf $W_i$ has. In other words, if $Y_i^q = \max\{Y_i\} > Y_i$, then $\boldsymbol{I}_i = \boldsymbol{I}_i^q$.

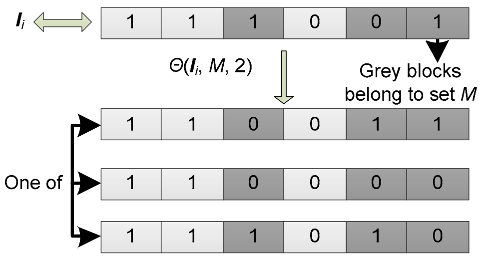


**Fig. 1 Illustration of the move operator**

4.1.5 Intelligent hunting behavior

Wolf leader $W_{\text{lead}}$ howls with position (item) $\boldsymbol{I}_{\text{lead}}(x)$ and resource $Y_{\text{lead}}$ of itself to gather other wolves around the prey resources. Artificial wolf $W_i$ moves toward wolf leader $W_{\text{lead}}$ with intelligent hunting step $\text{Step}_b$, during which the difference of the position (item) and resource between wolf leader $W_{\text{lead}}$ and artificial wolf $W_i$ is considered. Here, the $(k+1)^{\text{th}}$-generation artificial wolf $W_i$ is defined as

$$\boldsymbol{I}_i^{(k+1)} = \Theta\left(\boldsymbol{I}_i^{(k)}, M_i^{(k)}, \text{Step}_b\right), \tag{33}$$

where $\boldsymbol{I}_i^{(k)}$ is the $k^{\text{th}}$-generation artificial wolf $W_i$, $\text{Step}_b$ is the intelligent hunting step coefficient, and $M_i^{(k)}$ is the invert set {$s_1$, $s_2$, …, $s_m$} of the $k^{\text{th}}$-generation artificial wolf $W_i$, comprising a serial number of the different binary characters between wolf leader $W_{\text{lead}}$ and artificial wolf $W_i$.

$$s_i = \begin{cases} \text{null}, & x_{i,j} = x_{\text{lead},j}, \\ i, & x_{i,j} \neq x_{\text{lead},j}, \end{cases} \tag{34}$$

where "null" is a null value.

$M_i^{(k)}$ preserves the same part of items between wolf leader $W_{\text{lead}}$ and artificial wolf $W_i$ to retain the elite advantages. Let the different parts of items between wolf leader $W_{\text{lead}}$ and artificial wolf $W_i$ be included in the invert set to increase population advantages. If $M_i^{(k)}$ is an empty set, i.e., artificial wolf $W_i$ obtains the same position (item) as wolf leader $W_{\text{lead}}$, then artificial wolf $W_i$ moves randomly around its current position, i.e., random forging behavior $\boldsymbol{I}_i^{(k+1)} = \Theta\left(\boldsymbol{I}_i^{(k)}, M, 1\right)$.

The step of artificial wolf $W_i$ moving toward the wolf leader depends on the binary coding length of artificial wolf $m$ and the resource difference between wolf leader $W_{\text{lead}}$ and artificial wolf $W_i$. We proposed $\text{Step}_b$ to quantify the moving speed:

$$\text{Step}_b = \left\lfloor m \cdot \frac{Y_{\text{lead}} - Y_i}{\max Y_0 - \min Y_0} \right\rfloor + \text{Step}_{\min}. \tag{35}$$

Here, $\max Y_0$ is the upper resource limit, $\min Y_0$ is the lower resource limit, $\lfloor A \rfloor$ is the floor function whose value is the greatest integer less than or equal to $A$, and $\text{Step}_{\min}$ is the minimum number of steps that makes sense to the artificial wolf (we select 1 in 0–1 knapsack problems).

4.1.6 Upgrading

After scouting and intelligent hunting behaviors, the positions (items) of artificial wolves are determined. The resources (objective function value) of each wolf can be calculated. The artificial wolf with the greatest objective function value will defeat the old wolf leader and become the new wolf leader. The new wolf leader is kept in the next generation to maintain the advantages of the wolf population. $R$ artificial wolves of the other $n$–1 artificial wolves are randomly selected to die due to resource shortage, and baby artificial wolves (with the same number) are born to maintain the diversity of the group and ecological balance.

$R$ is a randomly selected integer in the interval $[n/(2\beta), n/\beta]$, where $\beta$ is the population upgrading coefficient.

### 4.2 IBWPA framework for UBKP

The framework of IBWPA is shown in Fig. 2. As can be seen, IBWPA applies wolf leader regulation to maintain elitism selection, applies the scouting behavior to find a better resource position (item) in local exploitation, applies the intelligent hunting behavior to improve search efficiency in global exploration, and applies upgrading to maintain population diversity during the search process. It emphasizes both the efficiency during the search process and the global features of the optimal solution.

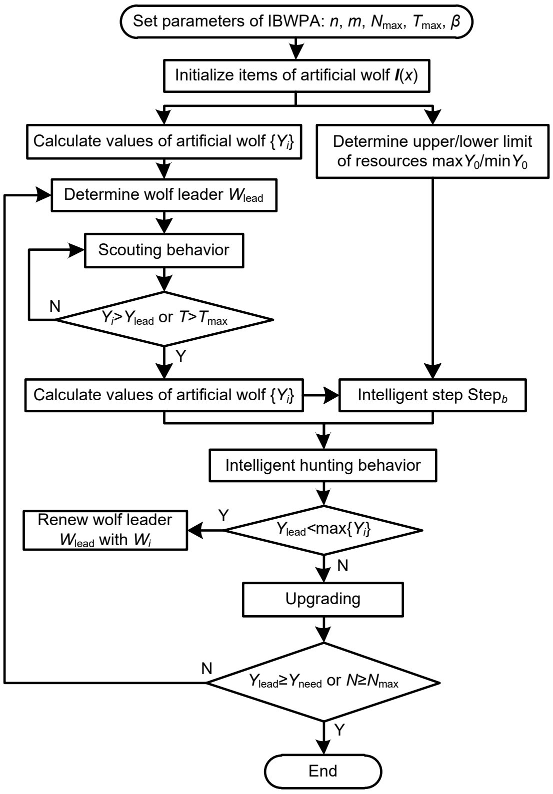


**Fig. 2 Framework of the improved binary wolf pack algorithm (IBWPA)**

Step 1: Set the parameters of IBWPA, the number of wolves $n$, the binary coding length of artificial wolf $m$, the maximum number of iterations $N_{\max}$, the maximum number of repetitions in scouting behavior $T_{\max}$, and the population upgrading proportional coefficient $\beta$.

Step 2: Initialize the position of artificial wolf $\boldsymbol{I}(x)$.

Step 3: Determine the $\mathcal{P}_E$ Nash equilibrium of followers of each solution item using IBWPA.

Step 4: Calculate the values of the leader knapsack for each solution item based on its $\mathcal{P}_E$ Nash equilibrium and select the wolf with the best resources (objective function value) as wolf leader $W_{\text{lead}}$.

Step 5: Every wolf $W_i$ performs the scouting behavior until $Y_i > Y_{\text{lead}}$ or the number of repetitions reaches $T_{\max}$.

Step 6: After the wolf leader $W_{\text{lead}}$ is determined, the remaining $n-1$ artificial wolves calculate the numbers of their intelligent steps and move toward the wolf leader.

Step 7: Upgrade the position (item) of the wolf leader according to wolf leader regulation and save it to the next generation. Then upgrade the wolf pack according to the population upgrading rule.

Step 8: If the value of the knapsack satisfies the requirement or the number of current iterations reaches the maximum number of iterations $N_{\max}$, the position (item) $\boldsymbol{I}_{\text{lead}}(x)$ and the resource (objective function value) $Y_{\text{lead}}$ of the wolf leader become the optimal solution of the knapsack problem; otherwise, go to step 3.

### 4.3 Analysis of IBWPA on classic KP benchmarks

To demonstrate the effectiveness of IBWPA in solving UBKP, simulations on classic KP benchmarks were performed because UBKP is a special type of KP.

To analyze the performance of IBWPA, five classical KP benchmarks with different dimensions (10–100) were selected. The parameters are shown in Table 1 (Liu JQ et al., 2007; Gao F et al., 2009; Qian and Zheng, 2012).

In this study, the maximum number of iterations $N_{\max}$ and the population size $n$ of IBWPA were set to the quadruple of knapsacks dimension $m$, i.e., $N_{\max}=n=4m$. These two parameters of the algorithms compared were the same as in IBWPA. The algorithms compared are the improved quantum genetic algorithm (IQGA) (Wang R et al., 2012), binary particle swarm optimization with time-varying acceleration coefficients (BPSOTVAC) (Chih et al., 2014), and the discrete cuckoo search (DCS) algorithm (Gherboudj et al., 2012). The binary coding length of the artificial wolf was the same as the dimension of the knapsack problem, the maximum number of repetitions in scouting behavior $T_{max}$ was set to 10, and the population upgrading proportional coefficient $\beta$ was set to 4. In each case, 100 independent tests of IBWPA with the above parameters were performed in MATLAB R2014a (8.0.3.532) using an Intel® Xeon® CPU E5-1620 v2@3.70 GHz running Windows 8.1 (64-bit).

**Table 1 Parameters of the five knapsack problem benchmarks**

| No. | Dimension | $V$ | $\Omega$ | $W$ | Best of known |
|---|---|---|---|---|---|
| KP1 | 10 | $V_1$ | $\Omega_1$ | 269 | 295-269 |
| KP2 | 20 | $V_2$ | $\Omega_2$ | 878 | 1024-871 |
| KP3 | 50 | $V_3$ | $\Omega_3$ | 1000 | 3119-1000 |
| KP4 | 80 | $V_4$ | $\Omega_4$ | 1173 | 5183-1170 |
| KP5 | 100 | $V_5$ | $\Omega_5$ | 3820 | 15 170-3818 |

$V$: value set; $\Omega$: weight set; $W$: weight limit. The best results of the known optimal solution are shown with the form of $v$-$w$ (value-weight)

To obtain the best performance with the algorithms compared, the best parameter settings for the IQGA, BPSOTVAC, and DCS were applied according to the literature (Gherboudj et al., 2012; Wang R et al., 2012; Chih et al., 2014). All the parameters are shown in Table 2.

**Table 2 Parameter settings adopted in the four different algorithms**

| Algorithm | Parameters |
|---|---|
| IBWPA | $n$=4$m$, $N_{max}$=4$m$, $T_{max}$=10, $\beta$=4 |
| IQGA | $n$=4$m$, $N_{max}$=4$m$, $a = b = 1/\sqrt{2}$ |
| BPSOTVAC | $n$=4$m$, $N_{max}$=4$m$, $c_1$=$c_2$=2, $\omega$=0.8 |
| DCS | $n$=4$m$, $N_{max}$=4$m$, $p_a$=0.3, $p_r$=0.3, $\beta$=1.5 |

The simulation results are shown in Table 3. Note that a searching test with an optimal solution greater than 99% of the best known optimal solutions was considered a successful test. The average convergence curves of the four algorithms for five typical knapsack benchmarks are shown in Figs. 3–7.

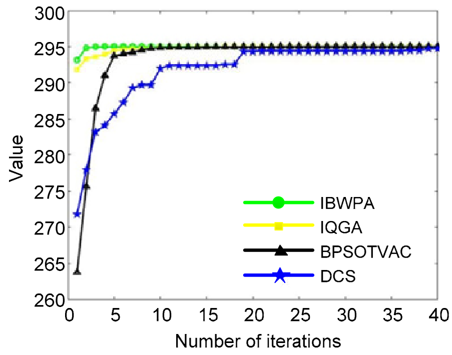


**Fig. 3 Average convergence curves for KP1**

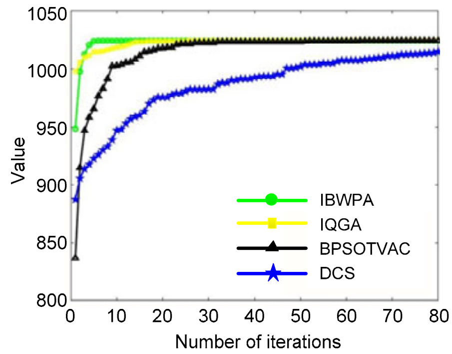


**Fig. 4 Average convergence curves for KP2**

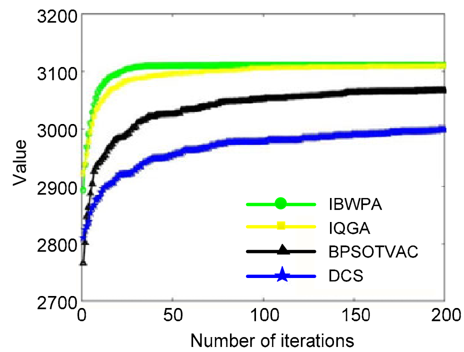


**Fig. 5 Average convergence curves for KP3**

From Table 3 and Figs. 3–7, we see that IBWPA obtained better optimal solutions in Best, Worst, and Avg. for all knapsack benchmarks, and its computation cost was the smallest due to the lowest AEN. These results demonstrate the robustness of IBWPA. IBWPA was also competitive compared to the best known optimal solutions because its SR value reached 100% on all KP benchmarks. The simulation results demonstrate that IBWPA is fast, efficient, and

**Table 3 Statistical results of 100 runs obtained by IBWPA, IQGA, BPSOTVAC, and DCS**

| No. | Best of known | Index | IBWPA | IQGA | BPSOTVAC | DCS |
|---|---|---|---|---|---|---|
| KP1 | 295-269 | Best | 295-269 | 295-269 | 295-269 | 295-269 |
| | | Worst | 295-269 | 295-269 | 295-269 | 293-249 |
| | | Avg. | 295 | 295 | 295 | 294.7 |
| | | SD | 0 | 0 | 0 | 0.57 |
| | | SR (%) | 100 | 100 | 100 | 100 |
| | | AEN | 1.2 | 1.9 | 3.5 | 6.25 |
| KP2 | 1024-871 | Best | 1024-871 | 1024-871 | 1024-871 | 1024-871 |
| | | Worst | 1024-871 | 1024-871 | 1024-871 | 986-726 |
| | | Avg. | 1024 | 1024 | 1024 | 1014.75 |
| | | SD | 0 | 0 | 0 | 9.39 |
| | | SR (%) | 100 | 100 | 100 | 65 |
| | | AEN | 2.6 | 6.7 | 19.45 | 62.95 |
| KP3 | 3119-1000 | Best | 3119-1000 | 3118-1000 | 3091-1000 | 3068-993 |
| | | Worst | 3112-1000 | 3102-1000 | 3044-998 | 2960-967 |
| | | Avg. | 3116 | 3110 | 3067 | 2994 |
| | | SD | 3.21 | 5.08 | 13.40 | 15.60 |
| | | SR (%) | 100 | 100 | 10 | 0 |
| | | AEN | 16.4 | 69.4 | 185.6 | 200 |
| KP4 | 5183-1170 | Best | 5183-1170 | 5183-1170 | 4877-1167 | 4691-1181 |
| | | Worst | 5183-1170 | 5088-1163 | 4643-1168 | 4463-1113 |
| | | Avg. | 5183 | 5155.15 | 4771.35 | 4552.15 |
| | | SD | 0 | 30.26 | 51.55 | 57.74 |
| | | SR (%) | 100 | 80 | 0 | 0 |
| | | AEN | 21.8 | 227.85 | 320 | 320 |
| KP5 | 15 170-3818 | Best | 15 170-3818 | 15 170-3818 | 14 349-3784 | 14 133-3370 |
| | | Worst | 15 170-3818 | 15 144-3809 | 14 030-3734 | 13 534-3212 |
| | | Avg. | 15 170 | 15 166.65 | 14 160.9 | 13 833.2 |
| | | SD | 0 | 6.23 | 90.95 | 157.61 |
| | | SR (%) | 100 | 100 | 0 | 0 |
| | | AEN | 24.35 | 74.35 | 400 | 400 |

Best, Worst, Avg., and SD stand for the best, worst, average, and standard deviation of 100 independent tests' optimal solutions, respectively. SR: success rate of tests; AEN: average evolutionary/iteration number. The best results of the known optimal solution are shown with the form of *v*-*w* (value-weight)

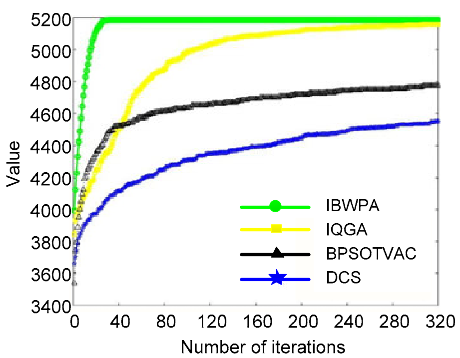


**Fig. 6 Average convergence curves for KP4**

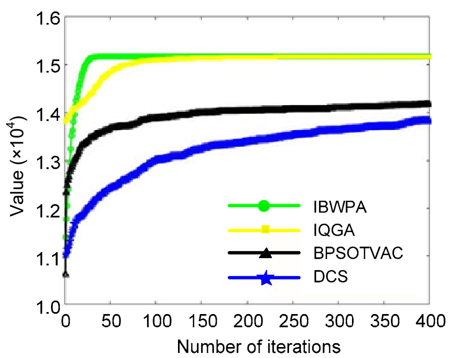


**Fig. 7 Average convergence curves for KP5**

robust in obtaining high-precision optimal solutions to the knapsack problem.

UBKP is a special knapsack problem with uncertain conditions; thus, we can expect that the proposed IBWPA can achieve good performance in solving UBKP.

## 5 Applications

This section presents an armament transportation example with uncertain parameters. In this example, several armaments and combat groups must be transported by airplane. In this problem, the frontier commander must maximize the combat capacity in the frontier, and the transportation commander attempts to guarantee safety in the transportation process. Assume that armaments and warrior soldiers provide two functions, i.e., combat capacity in the frontier and protective capacity in the transportation process. Then, an uncertain bilevel programming model of the armament transportation problem is given as follows:

$$\max_{(\boldsymbol{x},\boldsymbol{y})} F(\boldsymbol{x},\boldsymbol{y},\xi)=\xi_1\boldsymbol{d}^1\boldsymbol{x}+\xi_2\boldsymbol{d}^2\boldsymbol{y}$$
$$\text{s.t.}\begin{cases}\boldsymbol{x}\in\{0,1\}^3,\\ \max_{y} F(\boldsymbol{y},\xi)=\boldsymbol{\xi}_3\boldsymbol{d}^3\boldsymbol{y}\ \ \text{s.t.}\ \begin{cases}\boldsymbol{b}^1\boldsymbol{x}+\boldsymbol{b}^2\boldsymbol{y}\le c^2,\\ \boldsymbol{y}\in\{0,1\}^4,\end{cases}\end{cases} \tag{36}$$

where $\boldsymbol{x}=(x_i)_{1\times n}$ are armaments without protective capacity, $\boldsymbol{y}=(y_i)_{1\times n}$ are armaments with protective capacity, $\xi_1\boldsymbol{d}^1(\xi_2\boldsymbol{d}^2)$ is the combat capacity factor of armament $\boldsymbol{y}$, $\xi_3\boldsymbol{d}^3$ is the protective capacity factor of armament $\boldsymbol{y}$, and $F(\boldsymbol{x}, \boldsymbol{y}, \xi)$ is the total combat effectiveness. Here, the relevant data are $\xi_1=\mathcal{Z}(0.8, 0.9, 1.05)$, $\boldsymbol{d}^1=[3, 4, 7]$, $\xi_2=\mathcal{Z}(0.6, 0.8, 1.05)$, $\boldsymbol{d}^2=[5, 1, 2, 7]$, $\xi_3=\mathcal{L}(0.8, 1.5)$, $\boldsymbol{d}^3=[2, 2, 3, 4]$, $\boldsymbol{b}^1=[3, 1, 2]$, $\boldsymbol{b}^2=[1, 2, 1, 4]$, and $c^2=4$.

An uncertain solution based on IBWPA was applied to solve model (36). The results demonstrate that this example has a Stackelberg–Nash equilibrium $\boldsymbol{x}=[0, 1, 1]^{\mathrm{T}}$, $\boldsymbol{y}=[0, 0, 1, 0]^{\mathrm{T}}$ and the objective values of the leader and the follower are 16.43 and 3.52, respectively. In other words, armaments $x_2$, $x_3$ and group $y_3$ should be sent to the frontier, and the expected combat capacity and expected protective capacity are 16.43 and 3.52, respectively.

## 6 Conclusions

In this paper, we have proposed a UBKP model, which is a special type of BKP with uncertain variables. An uncertain solution has been developed to transform UBKP to an equivalent deterministic model. Computational efficiency has been improved by the proposed evolutionary IBWPA. The performance of IBWPA was much better than that of the algorithms compared for several typical knapsack problem benchmark tests. Finally, IBWPA has been applied to an armament transportation problem under uncertain parameters.

### Contributors

Ren-bin XIAO designed the research. Jun-jie XUE and Jin-qiang HU processed the data. Hu-sheng WU drafted the manuscript. Ren-bin XIAO helped organize the manuscript. Hu-sheng WU revised and finalized the paper.

### Compliance with ethics guidelines

Hu-sheng WU, Jun-jie XUE, Ren-bin XIAO, and Jin-qiang HU declare that they have no conflict of interest.

### References


Bai T, Wei J, Yang WW, et al., 2018. Multi-objective parameter estimation of improved Muskingum model by wolf pack algorithm and its application in upper Hanjiang River, China. *Water*, 10(10):1415. https://doi.org/10.3390/w10101415

Brotcorne L, Hanafi S, Mansi R, 2013. One-level reformulation of the bilevel knapsack problem using dynamic programming. *Dis Optim*, 10(1):1-10. https://doi.org/10.1016/j.disopt.2012.09.001

Caprara A, Carvalho M, Lodi A, et al., 2014. A study on the computational complexity of the bilevel knapsack problem. *SIAM J Optim*, 24(2):823-838. https://doi.org/10.1137/130906593

Carvalho M, Lodi A, Marcotte P, 2018. A polynomial algorithm for a continuous bilevel knapsack problem. *Oper Res Lett*, 46(2):185-188. https://doi.org/10.1016/j.orl.2017.12.009

Chih M, Lin CJ, Chern MS, et al., 2014. Particle swarm optimization with time-varying acceleration coefficients for the multidimensional knapsack problem. *Appl Math Model*, 38(4):1338-1350. https://doi.org/10.1016/j.apm.2013.08.009

Cohn AM, Barnhart C, 1998. The stochastic knapsack problem with random weights: a heuristic approach to robust transportation planning. Proc Triennial Symp on Transportation Analysis, p.1-13.

Dempe S, Richter K, 2000. Bilevel programming with knapsack constraints. *Centr Eur J Oper Res*, 8(2):93-107.

Gao F, Cui G, Wu ZB, et al., 2009. Virus-evolutionary particle swarm optimization algorithm for knapsack problem. *J Harbin Inst Technol*, 41(6):103-107 (in Chinese).

Gao YJ, Zhang FM, Zhao Y, et al., 2019. A novel quantum-inspired binary wolf pack algorithm for difficult knapsack problem. *Int J Wirel Mob Comput*, 16(3):222-232. https://doi.org/10.1504/IJWMC.2019.099861

Gherboudj A, Layeb A, Chikhi S, 2012. Solving 0-1 knapsack problems by a discrete binary version of cuckoo search algorithm. *Int J Bio-Inspir Comput*, 4(4):229-236. https://doi.org/10.1504/IJBIC.2012.048063

Han ZH, Tian XT, Ma XF, et al., 2018. Scheduling for re-entrant hybrid flowshop based on wolf pack algorithm. *IOP Conf Ser Mater Sci Eng*, 382(3):032007. https://doi.org/10.1088/1757-899X/382/3/032007

Li H, Zhang L, Jiao YC, 2014. Solution for integer linear bilevel programming problems using orthogonal genetic algorithm. *J Syst Eng Electron*, 25(3):443-451. https://doi.org/10.1109/JSEE.2014.00051

Li H, Xiao RB, Wu HS, 2016. Modelling for combat task allocation problem of aerial swarm and its solution using wolf pack algorithm. *Int J Innov Comput Appl*, 7(1): 50-59. https://doi.org/10.1504/IJICA.2016.075473

Liu B, 2009a. Some research problems in uncertainty theory. *J Uncert Syst*, 3(1):3-10.

Liu B, 2009b. Theory and Practice of Uncertain Programming. Springer Verlag, Berlin, Germany.

Liu B, 2010. Uncertainty Theory: a Branch of Mathematics for Modeling Human Uncertainty. Springer Verlag, Berlin, Germany.

Liu B, 2016. Uncertainty Theory ($5^{th}$ Ed.). Springer Verlag, Berlin, Germany.

Liu JQ, He YC, Gu QQ, 2007. Solving knapsack problem based on discrete particle swarm optimization. *Comput Eng Des*, 28(13):3189-3191, 3204 (in Chinese). https://doi.org/10.3969/j.issn.1000-7024.2007.13.052

Özaltın OY, Prokopyev OA, Schaefer AJ, 2010. The bilevel knapsack problem with stochastic right-hand sides. *Oper Res Lett*, 38(4):328-333. https://doi.org/10.1016/j.orl.2010.04.005

Qian J, Zheng JG, 2012. Greedy quantum-inspired evolutionary algorithm for quadratic knapsack problem. *Comput Integr Manuf Syst*, 18(9):2003-2011 (in Chinese).

Qiu X, Kern W, 2015. Improved approximation algorithms for a bilevel knapsack problem. *Theor Comput Sci*, 595(30): 120-129. https://doi.org/10.1016/j.tcs.2015.06.027

Wang R, Guo N, Xiang FH, et al., 2012. An improved quantum genetic algorithm with mutation and its application to 0-1 knapsack problem. Proc Int Conf on Measurement, Information and Control, p.484-488. https://doi.org/10.1109/MIC.2012.6273347

Wang ZT, Guo JS, Zheng MF, et al., 2015. A new approach for uncertain multiobjective programming problem based on $\mathcal{P}_E$ principle. *J Ind Manag Optim*, 11(1):13-26. https://doi.org/10.3934/jimo.2015.11.13

Wu HS, Zhang FM, Wu LS, 2013. New swarm intelligence algorithm—wolf pack algorithm. *Syst Eng Electron*, 35(11):2430-2438 (in Chinese).

Xue JJ, Wang Y, Li H, et al., 2016. A smart wolf pack algorithm and its convergence analysis. *Contr Dec*, 31(12): 2131-2139 (in Chinese). https://doi.org/10.13195/j.kzyjc.2015.1202