Partner selection model and soft computing approach for dynamic alliance of enterprises

WANG Dingwei (汪定伟)\(^1\), K. L. Yung (容启亮)\(^2\) & W. H. Ip (叶伟雄)\(^2\)

1. Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang 110015, China;  
2. Department of Manufacturing Engineering, The Hong Kong Polytechnic University, Hong Kong, China

Received July 26, 2001

Abstract  Partner selection is an active research topic in agile manufacturing and supply chain management. In this paper, the problem is described by a 0-1 integer programming with non-analytical objective function. Then, the solution space is reduced by defining the inefficient candidate. By using the fuzzy rule quantification method, a fuzzy logic based decision making approach for the project scheduling is proposed. We then develop a fuzzy decision embedded genetic algorithm. We compare the algorithm with traditional methods. The results show that the suggested approach can quickly achieve optimal solution for large size problems with high probability. The approach was applied to the partner selection problem of a coal fire power station construction project. The satisfactory results have been achieved.

Keywords: agile manufacturing, dynamic alliance, partner selection, soft computing, fuzzy logic, genetic algorithm.

To be competitive in the global manufacturing environment, the alliance with other enterprises becomes the most important factor\(^1\). “Opportunity Driven” operation manner based on bidding and tendering in Internet will be mainly adopted by dynamic alliances of enterprises\(^2\). When an enterprise, the main contractor, wins a bid, it usually has to sub-divide the won bid and sub-contract some portions of it out again by tendering and find the best partners to complete the task. Although there are many factors that affect partner selection such as friendship, credit and reliability, the key trade-off is always based upon the cost and completion time.

Partnership and partner selection have been widely researched into under agile manufacturing and supply chain management in recent years\(^3\). It is also an important function for information management systems of extended/virtual enterprises\(^4\). However, the mathematical models and optimization methods for partner selection are still a challenge to operations research\(^3\). Talluri et al. proposed a two-phase mathematical programming approach to the partner selection in the design of a virtual enterprise\(^5\). In our problem, the relationship of sub-projects contracted by partners may be represented by an activity network with precedence\(^6\). Thus, the problem could be considered as a partner selection problem with embedded project scheduling\(^7\). A further dimension to be considered being the investment for the project is usually paid to the main contractor by installments. The installments can be described as a cash flow that often cannot meet the payments to the sub-project contractors. Thus, the main contractor has to borrow from banks and pay interest to banks. To solve this combinatorial optimization problem of partner selection we developed a project scheduling embedded branch and bound algorithm. However, it cannot solve large...
size problems in an acceptable time due to the fatal weakness of enumeration of branch and bound.

Sine Zadeh first introduced the concepts of soft computing\(^8\), it has been an area of interest to many researchers. The basic idea of soft computing is to use the hybrid intelligent methods to quickly achieve an inexact solution rather than the exact optimal solution through long search\(^9\). Since genetic algorithms are good for adaptive search and fuzzy logic can be used to solve complex problems by linguistic rule-based techniques\(^10,11\), the combination of genetic algorithm and fuzzy logic become one of the most promising hybrid intelligent methods to solve complicated optimization problems. Wang et al. developed a genetic algorithm with fuzzy decision for the fuzzy due-date bargaining problem and generalized the idea to fuzzy rule quantification\(^12\).

To study partner selection problem, we describe it with a 0-1 integer programming with non-analytical objective function. By defining inefficient candidates the solution space can be reduced greatly. Then using the fuzzy rule quantification method, a fuzzy logic based decision making approach for the project scheduling is proposed. The approach was applied to the partner selection problem of a coal fire power station construction project. The satisfactory results have been achieved. Comparing the approach with branch and bound algorithm, the results show that the suggested approach can quickly achieve optimal solution for large size problems with high probability.

1 Model for partner selection

1.1 Problem definition

Assume an enterprise wins the bid of a big project consisting of several sub projects. To describe the sub projects more clearly, we call them jobs. The enterprise is not able to complete every aspect of the whole project by itself. Therefore, it has to call tender for these jobs. On the other hand, we can consider the enterprise to be a virtual company without any kinds of manufacturing resources\(^2\). The major strategy of this company is to win bids and distribute all jobs to other contractors through further tendering. The profit is the difference between the project investment and the sub project costs.

Let the project consist of \(n\) jobs. From the precedence relationship between these jobs, they form an activity network\(^6\). If job \(k\) can only begin after the completion of job \(i\), i.e. job \(i\) precedes job \(k\), we define the connected job pair by \((i, k) \in H\). There \(H\) is the set of all connected job pairs. For the convenience of description, we label these jobs such that \(i < k\), \(\forall (i, k) \in H\). Without loss of generality, the final job is labeled job \(n\). If the final job cannot be determined, a virtual final job can be created. The project owner will pay the main contractor by a cash flow note as \(e(t) \geq 0, t = 1, 2, \cdots, D\), where \(D\) is the due date of the project. If the project is tardy, the main contractor will be penalized by a tardiness penalty of \(\beta\) per tardy period.

For job \(i\), \(i = 1, 2, \cdots, n\), \(m\) candidates respond to the invitation for tender. For the candidate \(j\) of job, \(i\), its bid cost is \(b_{ij}\) and processing time is \(q_{ij}\) periods. To simplify the problem, we assume that the main contractor will pay the job cost to the selected sub-contractors by installments. The first payment is \(\alpha \times b_{ij} (0 < \alpha < 1)\) at the beginning of job \(i\) and the remainder \((1 - \alpha) \times b_{ij}\) upon completion of the job \(i\), \(i = 1, 2, \cdots, n\). In the case of money shortage, the main
contractor may obtain a loan from a bank with an interesting rate of \( r > 0 \).

The objective here is to select the best combination of partner enterprises for all jobs to minimize the total cost of the project by taking into consideration the job costs to all selected partners, the bank loan interest and the tardiness penalty.

Define variables

\[
w_{ij}(t) = \begin{cases} 
1, & \text{job } i \text{ is contracted to candidate } j \text{ and begins at period } t, \\
0, & \text{otherwise},
\end{cases} \quad \forall i, j, t. \tag{1}
\]

Then, the problem can be described by the following model:

\[
\min_{w} Z(w) = \sum_{i=1}^{n} \sum_{j=1}^{m} b_{ij} \sum_{t=1}^{\tau_{ij}} w_{ij}(t) + r \sum_{i=1}^{n} \sum_{j=1}^{m} \sum_{t=1}^{\tau_{ij}} b_{ij} w_{ij}(t)
\]

\[
+ (1 - a) \sum_{i=1}^{n} \sum_{j=1}^{m} \sum_{t=1}^{\tau_{ij}} b_{ij} w_{ij}(t - q_{ij}) - \sum_{\tau=1}^{\tau_{ij}} e(\tau) + \beta [c_{n} - D]^{+}, \tag{2}
\]

s.t.

\[
\sum_{j=1}^{m} \sum_{t=1}^{\tau_{ij}} w_{ij}(t) = 1, \quad i = 1, 2, \ldots, n, \tag{3}
\]

\[
\sum_{j=1}^{m} \sum_{t=1}^{\tau_{ij}} (t + q_{ij}) w_{ij}(t) \leq \sum_{j=1}^{m} \sum_{t=1}^{\tau_{ij}} t w_{ij}(t), \quad \forall (i, k) \in H, \tag{4}
\]

\[
\sum_{j=1}^{m} \sum_{t=1}^{\tau_{ij}} (t + q_{ij}) w_{ij}(t) = c_{n}, \tag{5}
\]

\[
w_{ij}(t) = 1 \text{ or } 0. \quad \forall i, j, t. \tag{6}
\]

where \([x]^{+}\) stands for \(\max\{0, x\}\).

Since the objective function is not continuous and differentiable, the model (2)—(6) cannot be solved by general mathematical programming approaches.

1.2 Sufficient condition for regularity

Although no earliness penalty can be found superficially in the objective function (2), it is a type of earliness and tardiness planning problems as long as the tardiness penalty cost \(\beta\) is large enough\[13,14\]. In fact the earliness penalty may be incurred in the earlier beginning of job that may lead to a higher interest cost due to possible loan. On the other hand, the project may be delayed as late as possible if the tardiness penalty is cheaper than the loan interest. In this case, we can only obtain a meaningless singular solution. Therefore, the first problem we consider is when the solution is not singular.

**Definition 1.** The problem (2)—(6) is regular if its objective function is of earliness and tardiness.

Let the total investment of the project, \(E\), be

\[
E = \sum_{t=1}^{\tau} e(t). \tag{7}
\]

Theorem 1 in the following gives a sufficient condition for regular solution.

**Theorem 1.** The partner selection problem (2)—(6) is regular if \(\beta \geq rE\).

**Proof.** To simplify the description, let \(a = 1\). The objective function \(Z(w)\) is rewritten as
\[
Z(w) = \sum_{i=1}^{r} \sum_{j=1}^{m} b_{ij} \sum_{\tau=1}^{d_i} w_j(t) + r \sum_{i=1}^{r} \left[ \sum_{j=1}^{m} \sum_{l=1}^{d_j} b_{ij} w_j(t) - \sum_{\tau=1}^{d_i} e(\tau) \right]^+ + \beta [c_n - D]^+.
\]

Assume \(w(t), t = 1, 2, \ldots, c_n\), is a solution of (2)—(6). Then, \(\bar{w}(t + 1) = w(t), t = 1, 2, \ldots, c_n\), is also a solution with one period of delay to \(w(t)\). It is clear that \(\bar{w}(t)\) can meet all constraints (3)—(6) and \(c_n = c_n + 1\). The difference of two objective values is

\[
Z(w) - Z(\bar{w}) = r \sum_{i=1}^{r} \left[ \sum_{j=1}^{m} \sum_{l=1}^{d_j} b_{ij} \bar{w}_j(t) - \sum_{\tau=1}^{d_i} e(\tau) \right]^+
- r \sum_{i=1}^{r} \left[ \sum_{j=1}^{m} \sum_{l=1}^{d_j} b_{ij} w_j(t) - \sum_{\tau=1}^{d_i} e(\tau) \right]^+
+ \beta [c_n - D]^+ - \beta [c_n - D]^+.
\]

\[
= r \sum_{i=1}^{r} \left[ \sum_{j=1}^{m} \sum_{l=1}^{d_j} b_{ij} w_j(t) - \sum_{\tau=1}^{d_i} e(\tau) \right]^+
- r \sum_{i=1}^{r} \left[ \sum_{j=1}^{m} \sum_{l=1}^{d_j} b_{ij} w_j(t) - \sum_{\tau=1}^{d_i} e(\tau) \right]^+
+ \beta [c_n - D]^+ - \beta [c_n - D + 1]^+
\]

\[
= r \sum_{i=1}^{r} \left[ \sum_{j=1}^{m} \sum_{l=1}^{d_j} b_{ij} w_j(t) - \sum_{\tau=1}^{d_i} e(\tau) \right]^+
- r \sum_{i=1}^{r} \left[ \sum_{j=1}^{m} \sum_{l=1}^{d_j} b_{ij} w_j(t) - \sum_{\tau=1}^{d_i} e(\tau) \right]^+
+ \beta [c_n - D]^+ - \beta [c_n - D + 1]^+.
\]

**Note.** Since \(w_j(0) = 0, \forall j, i,\) we can remove the second \([ \cdot ]^+\) when \(t = 0\), because the expression inside becomes zero for \(e(1) \geq 0\).

Define \(y(t) = \sum_{i=1}^{r} \sum_{j=1}^{m} b_{ij} w_j(t) - \sum_{\tau=1}^{d_i} e(\tau)\). Then

\[
Z(w) - Z(\bar{w}) = r \sum_{i=1}^{r} \left[ [y(t) + e(t + 1)]^+ - [y(t)]^+ \right] + \beta [c_n - D]^+ - \beta [c_n - D + 1]^+.
\]

**Case 1.** \(c_n < D\).

We have \([c_n - D]^+ = [c_n - D + 1]^+ = 0\). Then,

\[
Z(w) - Z(\bar{w}) \geq 0,
\]

because \([y(t) + e(t + 1)]^+ \geq [y(t)]^+ \forall t\).

**Case 2.** \(c_n \geq D\).

\[
Z(w) - Z(\bar{w}) \leq r \sum_{i=1}^{r} e(t + 1) - \beta \leq rE - \beta \leq 0.
\]
Theorem 1 is explicit in practice. For the investor, the profit per period of the project must be greater than the interest of total investment at least. And the tardiness penalty must be greater than the lost profit of the project for a single period at least. Therefore, the sufficient condition for regular solution mentioned in Theorem 1 can be easily obtained in practice.

2 Model simplification and encoding scheme

The size of the solution space (the number of feasible solutions) of problem (2)–(6), $N$, is

$$N = \prod_{i=1}^{m} m_i.$$  \hfill (9)

It is evident that the size is very large even for a small scale problem. For example, $N = 9.5367 \times 10^{13}$ for a problem with 20 jobs and 5 candidates for each job. Therefore, the reduction of solution space is very important.

**Definition 2.** The candidate $j$ of job $i$ is inefficient if there exists a candidate $k$ for the same job with $b_{ik} \leq b_{ij}$, $q_{ik} < q_{ij}$ or $b_{ik} < b_{ij}$, $q_{ik} \leq q_{ij}$.

**Theorem 2.** If problem (2)–(6) is regular, there exists at least one optimal solution not including any inefficient candidates.

**Proof.** Assume $w'(t), t = 1, 2, \cdots, c_n$, is an optimal solution with an inefficient candidate $j$ for job $i$. By Definition 2, there is a candidate $k$ of job $i$ with $b_{ik} \leq b_{ij}$, $q_{ik} < q_{ij}$ or $b_{ik} < b_{ij}$, $q_{ik} \leq q_{ij}$.

Replacing the candidate $j$ of job $i$ by $k$ of solution $w'(t), t = 1, 2, \cdots, c_n$, we get a new solution $w(t), t = 1, 2, \cdots, c_n$. It is easy to prove that $Z(w) \leq Z(w')$ as long as the problem (2)–(6) is regular.

Therefore, $w(t), t = 1, 2, \cdots, c_n$, is also an optimal solution, or the assumption is not true.

According to Theorem 2 we can ignore all inefficient candidates in the procedure to solve the problem (2)–(6) without the loss of optimal solution. Thus, the solution space can be reduced effectively.

We sequence the candidates for job $i$ on the bid cost from low to high. It is

$$b_{i1} < b_{i2} < \cdots < b_{in}, \ i = 1, 2, \cdots, n.$$  \hfill (10)

In the case that all inefficient candidates have been eliminated, we have

$$q_{i1} > q_{i2} > \cdots > q_{in}, \ i = 1, 2, \cdots, n.$$  \hfill (11)

**Note.** According to Definition 2, there will be no candidate pair with only one equality among the two inequalities of cost and processing time if all inefficient candidates are eliminated. The candidates with equal cost and processing time cannot be identified by this model. Therefore, they will be treated as one candidate in the model.

For the genetic algorithm, we take the natural number string as the gene representation. Let $x = [x_1, x_2, \cdots, x_n]$, where $x_i$ is an integer between 1 and $m_i$, $\forall i$. It means that the candidate $x_i$ is selected for the job $i$. Thus, $x = [x_1, x_2, \cdots, x_n]$ is referred to as a selection. A selection represents a chromosome or an individual in the genetic algorithm. For example, a natural
number string of 8 bits:
\[
\begin{bmatrix}
3 & 2 & 5 & 1 & 4 & 1 & 6 & 2
\end{bmatrix}
\]
is a selection or a chromosome of a project of 8 jobs. It means that candidate 3 for job 1, candidate 2 for job 2, and so on, are selected.

Let \( s_i(x) \) and \( c_i(x) \) be the beginning time and completion time of job \( i \) for the selection \( x \). Then, the model (2)–(6) is rewritten by

\[
\min Z(x) = \sum_{i=1}^{n} b_{i_1} + r \sum_{i=1}^{n} \left[ \alpha \sum_{i=1}^{n} b_{i_1} + (1 - \alpha) \sum_{i=1}^{n} c_{i_1} - \sum_{r=1}^{\tau_i} e(r) \right]^{+} + \beta[c_n(x) - D]^{+},
\]

s.t. \( c_i(x) \leq s_k(x), \quad \forall (i, k) \in H \),

\( x_i \) is the integer between 1 and \( m_i \), \( i = 1, 2, \ldots, n \).

We can see that the rewritten model (12)–(14) is much simpler than the original model (2)–(6), but there are some variables in its subscripts. This is difficult to be handled with the traditional mathematical modelling but easy with genetic algorithms.

For a pair of selections \( x \) and \( y \), \( x > y \) means that \( x_i \geq y_i \), \( \forall i \) and \( x_k > y_k \) for at least one existing \( k \) among 1 to \( n \).

Let the net cost of a selection, \( V(x) \), be the sum of job costs and the loan interest for the selection \( x \), i.e.

\[
V(x) = Z(x) - \beta[c_n(x) - D]^{+}.
\]

It means the tardiness penalty is not included in the net cost. And, let the increments of the job cost and processing time of the candidate \( h \) for job \( i \) be \( \Delta b_{ih} \) and \( \Delta q_{ih} \),

\[
\Delta b_{ih} = b_{(i,h+1)} - b_{ih}, \quad h = 1, 2, \ldots, m_i - 1, \quad i = 1, 2, \ldots, n,
\]

\[
\Delta q_{ih} = q_{(i,h+1)} - q_{ih}, \quad h = 1, 2, \ldots, m_i - 1, \quad i = 1, 2, \ldots, n.
\]

From inequalities (10) and (11), it is clear that

\[
\Delta b_{ih} > 0 \quad \text{and} \quad \Delta q_{ih} < 0, \quad \forall \ h, i.
\]

**Theorem 3.** For a pair of selections \( x \) and \( y \), if \( x > y \), then, \( c_n(x) \leq c_n(y) \) and \( V(x) \geq V(y) \) as long as

\[
\Delta b_{ih} + rb_{ih} \Delta q_{ih} \geq 0, \quad \forall \ h, i.
\]

**Proof.** Let \( x^2 \) and \( x^1 \) be two selections with \( x^2 > x^1 \) and \( x_i^1 - 1 = x_i^2 = h \) only for \( i \) and \( x_i^2 \neq x_i^1, \quad \forall k \neq i \). Because \( c_n(x) \) is equal to the sum of processing times of all jobs in critical path and the critical path is the longest path from starting to terminal nodes. We have

(i) It is easy to prove that \( c_n(x^2) \leq c_n(x^1) \), and the strict inequality holds as long as job \( i \) is in the critical path of selection \( x^1 \) or \( x^2 \).

(ii) Because the largest possible advancement of \( x^1 \) over \( x^2 \) for job \( i \) is \( q_{ih} - q_{i(i,h+1)} \), the largest possible increment in loan interest of \( x^1 \) over \( x^2 \) is \( rb_{ih}(q_{ih} - q_{i(i,h+1)}) \). Thus, we have

\[
V(x^1) - V(x^2) \leq b_{ih} - b_{i(i,h+1)} + rb_{ih}(q_{ih} - q_{i(i,h+1)}) = -\Delta b_{ih} - rb_{ih} \Delta q_{ih} \leq 0.
\]

Then, the theorem for \( x^2 > x^1 \) is proved. Starting with \( x^1 = y \), \( x^2 \) is set by adding one to these bits \( i \) with \( x_i > y_i \). Repeat the above proof and set \( x^1 \leftarrow x^2 \) until \( x^2 = x \). Then, the theorem is completed.
Since $r$ is usually very small, condition (19) is easily met for most practical problems.
There are two special selections in the solution space. They are defined as follows:

Definition 3. The selection $x_l = [1, 1, \cdots, 1]$ and $x_r = [m_1, m_2, \cdots, m_n]$ are defined as the latest and earliest selections respectively.

Corollary 1. If (19) is met, the latest selection $x_l$ has the longest completion time $c_n(x_l)$ and the lowest net cost $V(x_l)$, and the earliest selection $x_r$ has the shortest $c_n(x_r)$ and the highest $V(x_r)$.

Proof. Since for any selection $x$, we have $x_l \leq x \leq x_r$, from Theorem 3, the following inequalities are true:

$$c_n(x_l) \geq c_n(x) \geq c_n(x_r)$$

and

$$V(x_l) \leq V(x) \leq V(x_r).$$

Corollary 1 provides the foundation for describing the fuzzy factors by proper membership functions.

Once a selection $x$ fixes the candidates for all jobs, the project scheduling can be done by the general CPM (Critical Path Method) procedure. The objective function value for a selection, the critical jobs in the critical path and non-critical jobs can also be determined by the procedure. It is noted that the different selections of candidates would cause different critical paths.

3 Fuzzy decision rules

The basic idea of fuzzy rule quantification can be described as follows.[12] The knowledge and experience of an expert usually can be represented by a collection of "IF…THEN…" rules. The rules can be quantified by fuzzy logic relationship of fuzzy factors. Then, the adapted decision for any states can be obtained by the fuzzy operations that respond to fuzzy logic.

As we know, the randomly generated selections in genetic algorithm has little chance of being useful while people who worked on scheduling have some valuable experiences that can be used to improve the selections. For example, the cheaper candidates can be selected for non-critical jobs to reduce total cost, and the faster candidates can be selected for the critical jobs when the project is tardy. Thus, it is natural to use the above fuzzy rule quantification for solving the problem.

To start the quantification, we introduce several sets that will be used in the description of the factors and decisions.

Definition 4. The set of plus-able critical jobs, $S_{pc}$, the set of minus-able critical jobs, $S_{mc}$, and the set of minus-able non-critical jobs, $S_{nn}$ are defined by

$$S_{pc} = \{ i \mid i \in x_i < m_i \quad \text{and} \quad i \in S_e \},$$
$$S_{mc} = \{ i \mid i \in x_i > 1 \quad \text{and} \quad i \in S_e \},$$
$$S_{nn} = \{ i \mid i \in x_i > 1 \quad \text{and} \quad i \in S_n \},$$

respectively, where, $S_e$ and $S_{ne}$ stand for the sets of critical jobs and non-critical jobs.

There are six factors to be considered for the fuzzy decision:

Factor 1: The expensiveness situation of net cost of the selection in consideration.
Factor 2: The tardiness situation of the selection.
Factor 3: The earliness situation of the selection.
Factor 4: The number of plus-able critical jobs, $N_{pc}$.
Factor 5: The number of minus-able critical jobs, $N_{mc}$.
Factor 6: The number of minus-able non-critical jobs, $N_{mnc}$.

Here, the numbers $N_{pc}$, $N_{mc}$ and $N_{mnc}$ are the element numbers of sets $S_{pc}$, $S_{mc}$ and $S_{mnc}$.

The first three factors are fuzzy. Their membership functions are defined by the following formulas. The final three factors are deterministic integers represented by $N_{pc}$, $N_{mc}$ and $N_{mnc}$ respectively.

To clear the description, we define

$$V_1 = V(x_1) + \gamma_1(V(x) - V(x_1)),$$

$$V_2 = V(x_1) + \gamma_2(V(x) - V(x_1)),$$

$$C_1 = c_n(x_1) + \gamma_3(c_n(x_1) - c_n(x_1)),$$

$$C_2 = c_n(x_1) + \gamma_4(c_n(x_1) - c_n(x_1)),$$

where $\gamma_1$, $\gamma_2$, $\gamma_3$ and $\gamma_4$ are relaxation coefficients. Usually, $\gamma_1$ and $\gamma_3$ are set in the range 0.0—0.1, and $\gamma_2$ and $\gamma_4$ in the range 0.9—1.0.

The expensiveness situation of net cost is represented as the fuzzy number $\tilde{F}_1$ with membership function $\mu_{\tilde{F}_1}(x)$,

$$\mu_{\tilde{F}_1}(x) = \begin{cases} 1, & \text{if } x > V_2, \\ (x - V_1)/(V_2 - V_1), & \text{if } V_1 \leq x \leq V_2, \\ 0, & \text{if } x < V_2. \end{cases}$$

The tardiness situation is represented by the fuzzy number $\tilde{F}_2$ with membership function $\mu_{\tilde{F}_2}(x)$,

$$\mu_{\tilde{F}_2}(x) = \begin{cases} 1, & \text{if } x > C_2, \\ (x - D)/(C_2 - D), & \text{if } D \leq x \leq C_2, \\ 0, & \text{if } x < D. \end{cases}$$

Similarly, the earliness situation is represented by the fuzzy number $\tilde{F}_3$ with the membership function $\mu_{\tilde{F}_3}(x)$,

$$\mu_{\tilde{F}_3}(x) = \begin{cases} 1, & \text{if } x < C_1, \\ 1 - (D - x)/(D - C_1), & \text{if } C_1 \leq x \leq D, \\ 0, & \text{if } x > D. \end{cases}$$

Once the above factors are evaluated, we can make decisions based on them. The decisions are composed of the basic modification operations. Two basic modification operations are designed. They are (i) the plus one operation, $O_1$, is to replace the candidate $x_i$ by $x_i + 1$ for job $i$; (ii) the minus one operation, $O_2$, is to replace the candidate $x_i$ by $x_i - 1$ for job $i$.

Then we get the decision space that includes 4 fuzzy decisions. They are:

Decision 1: Randomly select $(1 - \mu_{\tilde{F}_1})N_{pc}$ critical jobs from $S_{pc}$ to do the plus one operation, and $(\mu_{\tilde{F}_2} + \mu_{\tilde{F}_3})N_{mnc}$ non-critical jobs from $S_{mnc}$ to do the minus one operation, where
Decision 2: Randomly select \( \lfloor \mu_{F}^c, \mu_{F}^n \rfloor \) critical jobs from \( S_{nc} \) and \( \lfloor \mu_{F}^c, \mu_{F}^n \rfloor \) non-critical jobs from \( S_{nnc} \) to do the minus one operation.

Decision 3: Randomly select \( \lfloor \mu_{F}^c(1 - \mu_{F}^n)(1 - \mu_{F}^n) \rfloor \) non-critical jobs from \( S_{nnc} \) to do the minus one operator.

Decision 4: Without any operation and return to main routing.

Decision 2 is to decrease the total cost by selecting cheaper candidates for both critical and non-critical jobs. Decision 3 is to decrease the net cost by changing only the non-critical jobs and to keep the critical jobs unchanged. The final decision is to keep all jobs unchanged.

Let \( \mu_{\tilde{D}_k} \) be the membership degree of fuzzy decision \( \tilde{D}_k \), \( k = 1, 2, 3, 4 \). Considering the effect of the three fuzzy factors on these decisions, we have:

Rule 1: \( \mu_{\tilde{D}_1} = \mu_{F} \otimes \mu_{F} \) \hspace{1cm} (30)

Rule 2: \( \mu_{\tilde{D}_2} = \mu_{F} \otimes \mu_{F} \) \hspace{1cm} (31)

Rule 3: \( \mu_{\tilde{D}_3} = \mu_{F} \otimes \mu_{F} \otimes \mu_{F} \) \hspace{1cm} (32)

Rule 4: \( \mu_{\tilde{D}_4} = \mu_{F} \otimes \mu_{F} \otimes \mu_{F} \) \hspace{1cm} (33)

where \( \mu^c = 1 - \mu \), and "\( \otimes \)" denotes Hamacher’s product, i.e.

\[
\mu_1 \otimes \mu_2 = \frac{\mu_1 \mu_2}{\delta + (1 - \delta)(\mu_1 + \mu_2 - \mu_1 \mu_2)},
\]

where \( \delta \) is a selected parameter for fuzzy product. Different fuzzy products can be obtained by selecting different values for \( \delta \). \[11\]

Given \( \mu_{\tilde{D}_1}, \mu_{\tilde{D}_2}, \cdots, \mu_{\tilde{D}_4} \), decision \( k \) is selected if

\[
k = \arg \max_{1 \leq k \leq 4} \{|\mu_{\tilde{D}_1}, \mu_{\tilde{D}_2}, \mu_{\tilde{D}_3}, \mu_{\tilde{D}_4}|\}.
\]

The above fuzzy logic operations may be interpreted as follows. (30) means that the choice of decision 1 is chosen if the project is tardy and its net cost is not high. (31) means that decision 2 is chosen in the case of the early completion and higher net cost. (32) means that decision 3 is chosen if the completion is just at due date and the net cost is higher. The meaning of (33) is clear, i.e. decision 4 is selected when the completion is just at due date and the net cost is also lower.

While the recommended method is intrinsically a rule-based method, it uses the fuzzy logic operations in (30)—(35) to determine the best decision instead of creating and activating a large number of rules as that with traditional rule-based methods.

4 Fuzzy decision embedded genetic algorithm

The basic idea of the fuzzy decision embedded genetic algorithm is to select the partner combination by genetic algorithm at the first level of algorithm, and to schedule all jobs with a fixed partner combination by fuzzy decision at the second level.

For our genetic algorithm, the natural number string specifying the partner selection of the \( n \)
jobs is taken as the gene representation. The fitness function is defined by linearly scaling the objective function of (12), i.e.
\[ f(j) = Z_{\text{max}} - Z(j) + a, \quad j = 1, 2, \ldots, NP, \]
where \( NP \) is the population size, \( Z_{\text{max}} = \max \{ Z(x_j), j = 1, 2, \ldots, NP \} \) is the maximum of the objective values achieved in the population, and \( a \) is a positive number which can be adjusted with the iteration process to change the selection pressure.

The two-cup-point crossover and the altering mutation are used as the genetic operators in the genetic algorithm. It is evident that all chromosomes can remain legal and feasible when the genetic operations of the above crossover and mutation are done. The commonly used "roulette wheel" with proportional selection is adopted as the selection strategy. Computations are stopped once a specified maximum number of generations has been examined.

The step by step procedure of the genetic algorithm embedded fuzzy decision is as follows:

Algorithm GA-FD:

Step 1. Specify the parameters: population size \( NP \), the maximum number of generations \( NG \), crossover probability \( p_c \), and mutation probability \( p_m \).

Step 2. Sequence and label the jobs to meet the inequalities (10) and (11).

Step 3. Call CPM procedure to calculate \( V(x_j), V(x_{\ast}), c_n(x_j) \) and \( c_n(x_{\ast}) \).

Step 4. Randomly generate an initial population with \( NP \) chromosomes.

\[ x(j) = [x_1(j), x_2(j), \ldots, x_n(j)], \quad x_i(j) \neq x_i(j), \quad \forall i, k; j = 1, 2, \ldots, NP, \]

and set the generation index \( k = 0 \).

Step 5. Let \( k = k + 1 \). If \( k > NG \) go to Step 10. Otherwise, do Steps 6 through 9.

Step 6. For chromosomes \( x(j), j = 1, 2, \ldots, NP \), do the following three sub steps:

Substep 6.1. Call CPM procedure to schedule all jobs and return the values of \( V(x(j)), c_n(x(j)), N_{\mu}(x(j)), N_{\mu}(x_{\ast}), V_{\mu}(x(j)), V_{\mu}(x_{\ast}) \).

Substep 6.2. Make the fuzzy decision and return the objective function value \( Z(j) \) and the modified selection \( x(j) \).

Substep 6.3. Find \( Z_{\text{max}} = \max \{ Z(j), j = 1, 2, \ldots, NP \} \) and \( Z_{\text{min}} = \min \{ Z(j), j = 1, 2, \ldots, NP \} \). \( j^* \) is the index of chromosome achieving \( Z_{\text{min}} \) and the associated partner selection.

Step 7. If \( Z_{\text{min}} < Z_{\ast} \), let \( Z_{\ast} = Z_{\text{max}} \) and \( x^* = x(j^*) \).

Step 8. For \( j = 1, 2, \ldots, NP \), calculate the fitness function for \( x(j) \) by formula (36). For \( j = 1, 2, \ldots, NP \), calculate the selection probability \( p(j) = f(j)/\sum_{j=1}^{NP} f(j) \).

Step 9. Do crossover and mutation with \( p_c \) and \( p_m \), update population.

Step 10. Output \( Z_{\ast} \) and \( x^* \) as the optimal solution.

5 Numerical analysis

An example of the problem is a real life problem of the construction project of a coal-fire power station.

The project consists of 16 jobs: 1) total design of system, 2) design of railway for coal, 3)
construction of railway, 4) design of boiler system, 5) manufacturing of boilers, 6) design of buildings, 7) construction of buildings, 8) design of generators, 9) design of electric transmission, 10) manufacturing of generators, 11) assembling of boilers, 12) manufacture of transmission equipment, 13) design and assembling of computer control system, 14) assembling of generators, 15) assembling of transmission system, 16) system inspection and test running. Its due date is 36 months. The tardiness penalty is RMB 48 million for one month. The payment rule for all jobs is 65 percent at the beginning of the job and the rest when the job is completed. The loan interest rate is 0.6 percent per month.

The precedence relationship represented by the Activity-on-Arc mode is shown in fig. 1.

![Fig. 1 The example of the node pair scheme.](image)

The total investment is RMB 485 million. The problem is shown to be regular due to 0.006 × 485 < 48.

The size of the solution space is 2.787 × 10^8. By Theorem 2, 9 inefficient candidates (P2) are identified and removed. The size is reduced considerably to 3.359 × 10^8. The remaining candidates satisfy the condition of (19).

By running the algorithm, the values for scaling the membership function are obtained:

V(x_l) = 397.00, V(x_r) = 549.60, e_u(x_l) = 53 and e_u(x_r) = 34.

The total cost of achieved solution is RMB 472,4842 million and the completion time is 36 and is just in time. The main contractor can obtain a profit of RMB 12.52 million from the project. Comparing the result with that by branch and bound algorithm, we see that the solution is optimal.

To test the performance of the fuzzy decision embedded genetic algorithm (GA/FD), we randomly produced some problems with different sizes. The results together with the comparison of the genetic algorithm without embedded fuzzy decision (GA) and the branch and bound algorithm (B&B) are shown in table 1. In the table, “Size” stands for the size of the reduced solution space, “CPU time” for the computation time of each running by CPU. The “Best rate” is tested by 100 running with different random seeds. The parameter settings for GA and GA/FD both are
\[ p_c = 0.8, \quad p_m = 0.1, \quad PN = 100 \quad \text{and} \quad NG = 1000. \]

**Table 1** The result comparison for the problems with different sizes

<table>
<thead>
<tr>
<th>n</th>
<th>Size</th>
<th>Alg.</th>
<th>CPU time</th>
<th>Best rate</th>
<th>Best</th>
<th>Mean</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>(3.359 \times 10^8)</td>
<td>GA/FD</td>
<td>14.55*</td>
<td>100%</td>
<td>472.48</td>
<td>472.48</td>
<td>472.48</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B&amp;B</td>
<td>1.32*</td>
<td>100%</td>
<td>472.48</td>
<td>472.48</td>
<td>472.48</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GA</td>
<td>8.82*</td>
<td>39%</td>
<td>472.48</td>
<td>473.71</td>
<td>479.30</td>
</tr>
<tr>
<td>22</td>
<td>(1.088 \times 10^{12})</td>
<td>GA/FD</td>
<td>20.27*</td>
<td>100%</td>
<td>678.42</td>
<td>678.42</td>
<td>678.42</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B&amp;B</td>
<td>15.71*</td>
<td>100%</td>
<td>678.42</td>
<td>678.42</td>
<td>678.42</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GA</td>
<td>11.99*</td>
<td>21%</td>
<td>678.42</td>
<td>682.03</td>
<td>694.10</td>
</tr>
<tr>
<td>27</td>
<td>(9.404 \times 10^{14})</td>
<td>GA/FD</td>
<td>22.54*</td>
<td>100%</td>
<td>917.68</td>
<td>917.68</td>
<td>917.68</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B&amp;B</td>
<td>14.61*</td>
<td>100%</td>
<td>917.68</td>
<td>917.68</td>
<td>917.68</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GA</td>
<td>13.39*</td>
<td>2%</td>
<td>918.67</td>
<td>928.22</td>
<td>940.85</td>
</tr>
<tr>
<td>30</td>
<td>(2.257 \times 10^{16})</td>
<td>GA/FD</td>
<td>26.24*</td>
<td>100%</td>
<td>1010.65</td>
<td>1010.65</td>
<td>1010.65</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B&amp;B</td>
<td>30.87*</td>
<td>100%</td>
<td>1010.65</td>
<td>1010.65</td>
<td>1010.65</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GA</td>
<td>15.16*</td>
<td>3%</td>
<td>1010.65</td>
<td>1020.46</td>
<td>1036.55</td>
</tr>
<tr>
<td>38</td>
<td>(3.510 \times 10^{20})</td>
<td>GA/FD</td>
<td>31.99*</td>
<td>80%</td>
<td>1362.47</td>
<td>1362.83</td>
<td>1366.14</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B&amp;B</td>
<td>458.47*</td>
<td>100%</td>
<td>1362.47</td>
<td>1362.47</td>
<td>1362.47</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GA</td>
<td>18.29*</td>
<td>1%</td>
<td>1362.47</td>
<td>1375.34</td>
<td>1388.98</td>
</tr>
<tr>
<td>48</td>
<td>(4.912 \times 10^{25})</td>
<td>GA/FD</td>
<td>40.52*</td>
<td>78%</td>
<td>1636.07</td>
<td>1636.55</td>
<td>1639.91</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B&amp;B</td>
<td>42457.73*</td>
<td>100%</td>
<td>1636.07</td>
<td>1636.07</td>
<td>1636.07</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GA</td>
<td>22.88*</td>
<td>1%</td>
<td>1639.55</td>
<td>1659.86</td>
<td>1682.51</td>
</tr>
<tr>
<td>60</td>
<td>(9.168 \times 10^{30})</td>
<td>GA/FD</td>
<td>49.31*</td>
<td>70%</td>
<td>1870.85</td>
<td>1871.61</td>
<td>1883.47</td>
</tr>
<tr>
<td></td>
<td></td>
<td>B&amp;B</td>
<td>142937.01*</td>
<td>100%</td>
<td>1870.85</td>
<td>1870.85</td>
<td>1870.85</td>
</tr>
<tr>
<td></td>
<td></td>
<td>GA</td>
<td>27.61*</td>
<td>1%</td>
<td>1899.27</td>
<td>1920.80</td>
<td>1946.12</td>
</tr>
</tbody>
</table>

From Table 1, we see that the problem size grows very fast with the job number. The recommended fuzzy decision embedded algorithm can achieve the optimal solution with a higher probability and the computation time does not grow fast with the size increase. The branch and bound algorithm can guarantee the optimum but it will take too long and too much memory to compute large scale problems. Although the genetic algorithm can solve larger problems faster, it cannot always find the optimal solution generally. The comparison strongly shows that the fuzzy decision is able to efficiently improve the computation performance of complex combinational optimization problems.

**Concluding remarks**

The work on the partner selection problem of extended enterprises leads to the following concluding remarks:

- The mathematical model (2) — (6) and its modified version (12) — (14) provide a formal description for the partner selection problems of dynamic alliance.
- The concept and theorem on the inefficient candidates provide the theoretical foundation for
reducing the solution space.

- The recommended genetic algorithm embedded fuzzy decision can quickly achieve the optimal solution of the mentioned problems with high probability. Comparing this with the genetic algorithm without fuzzy decision and the branch and bound algorithms, it has better performance in both computational speed and optimality. The computation results show its potential in practical partner selection and sub project management problems.

Acknowledgements This work was partly supported by the National Natural Science Foundation of China (Grant Nos. 69974039, 60084003) and the National High-Tech Program (Grant No. 863-511-9844-011) of China, and partly by the Hong Kong Polytechnic University research grant (G-YB95).

References