A heuristic for a special kind of multi-type component assignment problem

Y Shi, Z Guan and S Qiu
School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai 200240, China

Abstract. The multi-type component assignment problem (MCAP) is to assign more than one type of components to positions in the system to maximize system reliability. This paper proposes a two-stage Birnbaum importance-based local search heuristic to address a special kind of MCAP where positions in the system can accept up to two types of components. To evaluate the performance of the heuristic, comprehensive simulation experiments are conducted across different types of systems. The results demonstrate the heuristic's efficiency and applicability. It shows that the Birnbaum importance is also promising in solving MCAPs.

1. Introduction
The reliability of a system is usually dependent upon its components. Under certain circumstances, some components with similar functions can be assigned to different positions in the system. The reliability of the system changes as the arrangement of components changes. The component assignment problem (CAP) is to find the optimal assignment of components that maximizes the reliability of the system. In practice, however, not all of the components in a system is exchangeable. Some positions in the system can only accept one type of components, while others can be assigned with two or more types of components. The CAP under such kind of restrictions is called the multi-type component assignment problem (MCAP).

The single-type component assignment problem (SCAP) is an NP-hard problem. The exact optimal assignment of a SCAP can only be obtained by the enumeration method, costing a large amount of time. A series of heuristics based on Birnbaum importance (BI) is proposed in order to search for suboptimal assignments in limited amount of time. The BI of position $i$ is the probability that the system failure is caused by the component failure in position $i$, which is defined as:

$$BI(i) = \frac{\partial R(p_1, p_2, ..., p_n)}{\partial p_j} = R(p_1, ..., p_{i-1}, 1, p_{i+1}, ..., p_n) - R(p_1, ..., p_{i-1}, 0, p_{i+1}, ..., p_n)$$

where $p_i$ denotes the reliability of component in position $i$ for $i = 1, 2, ..., n$ in a system with $n$ components. $R(\cdot)$ denotes the reliability function of the system corresponding to the reliability of components in their respective positions. For heuristics applied to SCAP, BI can be seen as a guiding variable of assignment initialization and rearrangement, hence improving the efficiency. Zuo and Kuo [1] proposed the ZK heuristics for linear consecutive $k$-out-of-$n$ (Lin/Con/$k$/n) systems. In a Lin/Con/$k$/n: $F(G)$ system, $n$ components are connected in an ordered sequence, and the system fails (works) if and only if $k$ consecutive components fail (work). The ZK heuristics iteratively exchange components in a pairwise way such that components with higher reliability are assigned to
positions with larger value of BI. Lin and Kuo [2] designed the LK heuristics, which initialize all positions with the least reliable component, then iteratively assign the remaining most reliable component to the position with the largest value of BI. Yao et al. [3] integrated these two heuristics and proposed the Birnbaum importance-based two-stage approach (BITA) that initializes the system with the LK heuristics and further improves it with the ZK heuristics. The BITA is the most effective local search method for SCAP. To break local optimum, BI-based genetic algorithms have been developed. Yao et al. [4] proposed a BI-based genetic local search (BIGLS) algorithm featuring a BI-based three-way exchange local search method to boost chromosomes. Cai et al. [5]’s BI-based genetic algorithm (BIGA) uses the second stage of BITA as the local search method instead on Lin et al’s [6] systems.

Very limited studies have been conducted on the MCAP. Zhu et al. [6] first formally defined and studied a specific form of MCAP, where each type of positions in the system can only be assigned to one type of components. To extend this definition, we consider that some positions with higher compatibility can accept up to two types of components. This more universal model of MCAP has not been studied yet, to the best of our knowledge.

In the MCAP, the total number of available components can be larger than the total number of system positions. In our model, it is assumed that the number of positions in the studied system equals the number of available components, denoted by \( n \). The number of types of components is denoted by \( m \).

For \( \alpha, \beta = 1, 2, \ldots, m \) (\( \alpha \neq \beta \)), the set of type \( \alpha \) components is denoted by \( C_\alpha \), while the set of positions that can only accept components in \( C_\alpha \) is denoted by \( P_\alpha \). \( P_\alpha \cap P_\beta = \emptyset \), \( C_\alpha \cup P_\beta = \Phi \). For \( \alpha, \beta = 1, 2, \ldots, m \) (\( \alpha \neq \beta \)), \( P_{\alpha, \alpha+1} \) denotes the set of positions that can accept components in \( C_\alpha \cup C_\beta \). \( P_{\alpha, \alpha+1} \cap P_{\beta, \beta+1} = \emptyset \).

In addition, for \( \alpha = 1, 2, \ldots, m \) and \( \beta = 1, 2, \ldots, m-1 \), \( P_\alpha \cap P_{\beta, \beta+1} = \emptyset \).

Naturally, for \( \alpha = 1, 2, \ldots, m \),
\[
|P_\alpha| < |C_\alpha| < |P_{\alpha-1, \alpha} \cup P_\alpha \cup P_{\alpha, \alpha+1}|
\]

is a necessary condition for the MCAP to be feasible, where we manually define \( P_{0,1} = P_{m,m+1} = \emptyset \).

The structure of the remainder of this paper is as follows. Section 2 discusses the procedures of our algorithm in detail. Section 3 evaluates our algorithm through simulation experiments. Section 4 gives conclusions and future research.

2. The procedures of the BITA-ZK heuristic

We extend the BITA and an alternative of the ZK heuristics (ZKD) to address our MCAP. The BITA-ZK heuristic is a two-stage method that generates a feasible solution in the first stage, then improves the system reliability by local searching in the second stage. The BITA generates two solutions with LKA and LKB heuristics, improves them with either ZKB or ZKD heuristics, and finally chooses the better one as the final result. The ZKD heuristic starts with the most reliable component, iteratively compares it with the position of the highest BI among all components less reliable than it, and performs exchange if the match between the more reliable component and the higher BI position improves system reliability. Readers should refer to Zuo and Kuo [7] and Yao et al. [3] for details.

The procedures of the BITA-ZK heuristic are as follows:

Stage 1. Generate a Feasible Solution.

Step 1.1. Initialize all positions in the system with an auxiliary component, which has equal reliability to the least (or most) reliable available components.

Step 1.2. Do loop, for \( i = 1, 2, \ldots, m-1 \):

1.2.1. If \( i = 1 \), select \( P_i^C = P_1 \), \( P_{i,2} \), \( P_2 \) as input positions. Otherwise, select \( P_i^C = \{ p \mid \text{The component on } p \in C_i \} \), \( P_{i,\alpha+1} \), \( P_{i+1} \) as input positions.
1.2.2. Select components in \( iC_i \) and \( C_{i+1} \) as input components, which has an equal number of input positions. Choose components in \( C_i \) as priority and fill up with components in \( C_{i+1} \) arbitrarily.

1.2.3. Assign input components to input positions by BITA (ignoring multi-type restrictions).

1.2.4. Exchange wrong components that violate multi-type restrictions until the assignment is feasible. The exchange rules are: 1) Start from the least (or most) reliable wrong component. 2) Always exchange it with an available component with the closest reliability to it. 3) Exchange wrong components between \( P_i^C \) and \( P_{i+1}^C \) first. When \( P_i^C \) has remaining wrong components while \( P_{i+1}^C \) has not (vice versa), exchange them with available components in \( P_{i+1}^C \) instead.

Stage 2. ZK Local Search

Step 2.1. Do loop, for \( i=1,2,...,m \)

2.1.1. Perform the ZKD heuristic on \( P_i^C \) with components on \( P_i^C \).

Step 2.2. Do loop, for \( i=1,2,...,m-1 \)

2.2.1. Perform the ZKD heuristic on \( P_{i+1}^C \) with components on \( P_{i+1}^C \).

Step 2.3. If the assignment has ever changed during Step 2.1 or Step 2.2, go back to Step 2.1. Otherwise, stop.

3. Simulation experiments

3.1. Design of Experiments

To evaluate the performance of the BITA-ZK heuristic on solving MCAP, we conduct numerical experiments in various systems associated with components of different reliability patterns. We design four coherent systems (See figure 1), eleven \( \text{Lin} / \text{Con} / k / n : G \) systems and eight \( \text{Lin} / \text{Con} / k / n : F \) systems, specifying multi-type restrictions for each system respectively. The size of the systems ranges from 5 to 60. Also, we consider components of three reliability patterns, namely highly, poorly and arbitrarily reliable components, whose reliabilities are randomly generated from uniform distribution on \([0.8,0.99]\), \([0.01,0.2]\), and \([0.01,0.99]\) respectively. Each component’s type is specified according to the related system’s size and multi-type restrictions, under the constraint of equation (2). To avoid biased results, for each trial corresponding to a system and a pattern of component reliability, 100 instances are generated by populating 100 sets of components reliabilities according to their reliability pattern.

In small systems, it is possible to find the exact optimal and worst assignment by the enumeration method in limited amount of time. But in large systems, the enumeration method is not implementable because it is too time-consuming and the memory required is insufficient. Therefore, we use the randomization method in large systems. For each instance, 10 000 random feasible assignments are generated, among which the best and worst reliable assignments are saved. In order to evaluate the quality of the solution generated by the heuristic, we calculate the standardized system reliability (SSR):

\[
SSR = \frac{R_{\text{opt}} - R_{\text{best}}}{R_{\text{opt}} - R_{\text{worst}}}
\]

where \( R_{\text{best}} \) denotes the reliability of assignment generated by the heuristic, \( R_{\text{opt}} \) and \( R_{\text{worst}} \) denote the reliability of the best and worst assignment generated by the enumeration method or the randomization method. A trial containing 100 instances generates 100 SSRs consequently. The Mean SSR (MSSR) is calculated to evaluate the performance of the heuristic on each trial.

In the remainder of this section, we first give an illustrative example to show how the BITA-ZK heuristic obtain its solution step by step. Then we evaluate its performance on other ten small systems.
compared with the enumeration method. Eventually, we evaluate its performance on twelve larger systems compared with the randomization method.

3.2. An illustrative example

Consider a MCAP on a $Lin/Con/7/10:G$ system. There are three types of components, denoted by type X, Y and Z for the convenience of illustration. Detailed configurations are shown in table 1 and table 2.

We now perform the BITA-ZK heuristic on this instance. The process of how the assignment evolves during the execution of the BITA-ZK heuristic is shown below in table 3. The second column of table 3 demonstrates the reliability assignment vector in the current step. We use symbol $|$ to partition different types of positions. If a component fits the multi-type restrictions, its value of reliability is **bolded** in the vector.

In Step 1.1, initialize all positions in the system with auxiliary components whose reliability all equal the least reliable component ($c_i = 0.1$). The current reliability assignment vector becomes $[0.1 0.1 | 0.1 0.1 0.1 | 0.1 | 0.1 0.1 0.1 | 0.1]$. Auxiliary components are used only to calculate the system reliability and Birnbaum importance in the following steps. They should never be considered as input components.

**Figure 1.** Diagrams of the four coherent systems
Table 1. Positions configurations.

| Set of positions | \( P_X \) | \( P_{X,Y} \) | \( P_Y \) | \( P_{Y,Z} \) | \( P_Z \) |
|------------------|---------|-------------|---------|-------------|---------|
| Members          | \{ p_1, p_2 \} | \{ p_3, p_4, p_5 \} | \{ p_6 \} | \{ p_7, p_8, p_9 \} | \{ p_{10} \} |

Table 2. Components configurations.

| Set of components | \( C_X \) | \( C_Y \) | \( C_Z \) |
|-------------------|---------|---------|---------|
| Members           | \{ c_1, c_2, c_3 \} | \{ c_4, c_5, c_6 \} | \{ c_7, c_8, c_9, c_{10} \} |
| Reliability       | 0.1 0.2 0.3 | 0.4 0.5 0.6 | 0.8 0.85 0.9 0.95 |

Table 3. The procedure of the BITA-ZK heuristic on an illustrative example

| Step              | [\( P_X \mid P_{X,Y} \mid P_Y \mid P_{Y,Z} \mid P_Z \)] | \( P_X^C \) | \( P_Y^C \) | \( P_Z^C \) |
|-------------------|--------------------------------------------------------|-----------|-----------|-----------|
| 1.1               | \([0.1 \ 0.1 \mid 0.1 \ 0.1 \mid 0.1 \mid 0.1 \mid 0.1] \) | \( \Phi \) | \( \Phi \) | \( \Phi \) |
| 1.2.3(1st iter.)  | \([0.1 \ 0.2 \mid 0.3 \ 0.4 \ 0.5 \mid 0.6 \mid 0.1 \ 0.1 \ 0.1 \mid 0.1] \) | \{ p_1, p_2, p_3 \} | \{ p_4, p_5, p_6 \} | \( \Phi \) |
| 1.2.3(2nd iter.)  | \([0.1 \ 0.2 \mid 0.3 \ 0.8 \ 0.85 \mid 0.9 \mid 0.95 \ 0.6 \ 0.5 \mid 0.4] \) | \{ p_1, p_2, p_3 \} | \{ p_4, p_5 \} | \{ p_7, p_8, p_9 \} |
| 1.2.4(2nd iter.)  | \([0.1 \ 0.2 \mid 0.3 \ 0.4 \ 0.6 \mid 0.5 \mid 0.95 \ 0.85 \ 0.9 \mid 0.8] \) | \{ p_1, p_2, p_3 \} | \{ p_4, p_5 \} | \{ p_7, p_8 \} |

In the first iteration of Step 2.2, select \( P_X, P_{X,Y} \) and \( P_Y \) as input positions, which are \( p_1 \) to \( p_6 \). Select equal number of components from \( C_X \) and \( C_Y \), which are \( c_i \) to \( c_6 \). Perform the BITA on input positions with input components, and we get \([0.1 \ 0.2 \mid 0.3 \ 0.4 \ 0.5 \mid 0.6 \mid 0.1 \ 0.1 \ 0.1 \mid 0.1] \). Luckily, all input components are feasible on input positions, so we skip Step 1.2.4 and head into the second iteration.

In the second iteration of Step 2.2, select \( P_Y^C = \{ p \mid \text{The component on } p \in C_Y \} \), \( P_{Y,Z} \) and \( P_Z \), which are \( p_4 \) to \( p_{10} \). Select equal number of components from \( C_Y \) and \( C_Z \), which are \( c_4 \) to \( c_{10} \). Perform the BITA on input positions with input components, and we get the current reliability assignment vector of \([0.1 \ 0.2 \mid 0.3 \ 0.8 \ 0.85 \mid 0.9 \mid 0.95 \ 0.6 \ 0.5 \mid 0.4] \). Components on \( p_4, p_5, p_6 \) and \( p_{10} \) are infeasible according to the multi-type restrictions.

In Step 1.2.4 we exchange components to make this assignment feasible. First, we exchange components on \( p_4 \) and \( p_{10} \). Since \( P_Z \) has no remaining wrong components now, we exchange components on \( p_5 \) and \( p_6 \), then \( p_6 \) and \( p_8 \), subsequently. Finally, we end the first stage with the feasible assignment \([0.1 \ 0.2 \mid 0.3 \ 0.4 \ 0.6 \mid 0.5 \mid 0.95 \ 0.85 \ 0.9 \mid 0.8] \).

Now we head into Stage 2. In Step 2.1, perform the ZKD heuristic on \( p_1 \) to \( p_3 \), \( p_4 \) to \( p_6 \) and \( p_7 \) to \( p_{10} \) subsequently. In Step 2.2, perform the ZKD heuristic on \( p_3 \) to \( p_5 \), \( p_7 \) to \( p_9 \) subsequently. Luckily, there’s no exchange on either of the two steps, so the whole procedure ends with the final assignment \([0.1 \ 0.2 \mid 0.3 \ 0.4 \ 0.6 \mid 0.5 \mid 0.95 \ 0.85 \ 0.9 \mid 0.8] \) (If any exchange occurs, we should go back to Step 2.1). The reliability of the system under this assignment is 0.0760. By performing the enumeration method, we find a same optimal assignment, which means SSR=1 for this instance.

3.3. Numerical experiments

3.3.1. Experiments on small systems. We further conduct trials on another ten small systems. There are two types of components for each system to accept. In these ten trials we also calculate “Achieve” representing the number of instances where the BITA-ZK heuristic can achieve the exact optimal
Table 4. Overall performance of the BITA-ZK heuristic on ten small systems.

| System   | Number of component types | Highly reliable components | Poorly reliable components | Arbitrarily reliable components |
|----------|---------------------------|----------------------------|---------------------------|---------------------------------|
|          | MSSR (Achieve) | Time/s | MSSR (Achieve) | Time/s | MSSR (Achieve) | Time/s |
| C1       | 2             | 0.9975(87) | 0.766 | 1.0000(100) | 0.765 | 0.9884(86) | 0.651 |
| C2       | 2             | 0.9827(73) | 0.574 | 0.9676(66) | 0.591 | 0.9692(70) | 0.620 |
| C3       | 2             | 0.9867(53) | 2.117 | 0.9900(57) | 1.904 | 0.9851(46) | 1.926 |
| C4       | 2             | 0.9969(88) | 1.158 | 0.9932(69) | 1.410 | 0.9941(82) | 1.408 |
| Lin Con 2/7:G | 2             | 0.9950(40) | 1.060 | 0.9937(59) | 1.215 | 0.9903(27) | 0.958 |
| Lin Con 2/8:G | 2             | 0.9865(36) | 1.855 | 0.9929(69) | 1.480 | 0.9822(24) | 1.431 |
| Lin Con 3/7:G | 2             | 0.9955(46) | 1.214 | 0.9897(75) | 1.439 | 0.9729(44) | 1.100 |
| Lin Con 3/8:G | 2             | 0.9937(55) | 1.353 | 0.9972(79) | 1.459 | 0.9856(56) | 1.694 |
| Lin Con 3/7:F | 2             | 0.9943(52) | 1.097 | 0.9994(54) | 1.181 | 0.9936(50) | 1.204 |
| Lin Con 3/8:F | 2             | 0.9954(56) | 1.555 | 0.9972(80) | 1.572 | 0.9957(58) | 1.753 |

assignment obtained by the enumeration method. The MSSR, Achieve and computation time for each trial are shown in table 4.

From table 4 we can conclude that the BITA-ZK heuristic has high accuracy and efficiency on small systems. In most trials the MSSRs obtained are larger than 0.98. In more than half of the 3000 instances, the BITA-ZK heuristic obtained the optimal assignment. Also, the BITA-ZK heuristic greatly saves computation time compared with the enumeration method. For example, performing BITA-ZK heuristic on Lin Con 2/8:G with highly reliable components in 100 instances only costs 1.855 seconds, which is only 4% of what the enumeration method costs (456.952 seconds).

3.3.2. Experiments on large systems. We now focus on situations when the system size is relatively large. As mentioned before, we use the randomization method instead of the enumeration method to calculate the SSR. Therefore, SSR can be larger than 1 under this circumstance. The number of types of components range from 3 to 5 in these trials. The MSSR, and computation time for each trial are shown in table 5.

Table 5. Overall performance of the BITA-ZK heuristic on twelve large systems.

| System       | Number of component types | Highly reliable components | Poorly reliable components | Arbitrarily reliable components |
|--------------|---------------------------|----------------------------|---------------------------|---------------------------------|
|              | MSSR | Time/s | MSSR | Time/s | MSSR | Time/s |
| Lin Con 10/20:G | 3     | 1.1098 | 4.374 | 1.1613 | 4.446 | 1.1003 | 4.590 |
| Lin Con 15/20:G | 3     | 1.0100 | 5.049 | 1.0976 | 4.975 | 1.088 | 4.643 |
| Lin Con 30/40:G | 4     | 1.1832 | 20.316 | 1.5011 | 20.580 | 1.4503 | 15.994 |
| Lin Con 35/40:G | 4     | 1.0360 | 18.831 | 1.1450 | 16.133 | 1.1347 | 15.884 |
| Lin Con 50/60:G | 5     | 1.5792 | 53.760 | 5.8066 | 46.937 | 8.2000 | 41.146 |
| Lin Con 55/60:G | 5     | 1.1074 | 48.145 | 1.4822 | 47.753 | 1.4983 | 39.753 |
| Lin Con 10/20:F | 3     | 1.0749 | 3.895 | 0.9927 | 3.619 | 0.9959 | 3.668 |
| Lin Con 15/20:F | 3     | 0.9715 | 3.564 | 0.9810 | 3.243 | 0.9964 | 3.485 |
| Lin Con 30/40:F | 4     | 1.1046 | 11.208 | 1 | 10.212 | 0.9985 | 11.797 |
| Lin Con 35/40:F | 4     | 1.0112 | 10.053 | 1 | 9.593 | 0.9977 | 10.766 |
| Lin Con 50/60:F | 5     | 1.1922 | 30.387 | 1 | 24.938 | 1 | 25.675 |
| Lin Con 55/60:F | 5     | 1.0383 | 23.414 | 1 | 23.801 | 1 | 23.065 |
From table 5 we can conclude that the BITA-ZK heuristic outperforms the randomization method in general. In most trials the MSSRs obtained are larger than 1, which means BITA-ZK heuristic has higher accuracy than the randomization method. In some trials the MSSR obtained are dramatically higher than 1 (8.2). That means when the number of simulations is not large enough for large systems in the randomization method, low quality results could be generated. As for the BITA-ZK heuristic, the performances are much more stable. On rare occasions when the randomization method achieves a better assignment, the BITA-ZK heuristic still remarkably saves the computation time. For instance, in the trial associated with an Lin/Con/30/40: F and arbitrarily reliable components, the MSSR is slightly less than 1(0.9985). But the computation time costed by 100 instances is 11.797 seconds, which is less than 1% of what the randomization method costs (1630.350 seconds). It should be noticed that in trials associated with poorly reliable components in the Lin/Con/30/40: F, Lin/Con/35/40: F, Lin/Con/50/60: F and Lin/Con/55/60: F system, as well as arbitrarily reliable components in the Lin/Con/50/60: F and Lin/Con/55/60: F system, the differences of reliability between the best and worst assignment obtained by the randomization method are less than 10^{-15}. So, their MSSRs are recorded as 1, representing that the BITA-ZK heuristic has the same performance as the randomization method in these trials.

4. Conclusion and further research
This paper proposes the BITA-ZK heuristic to address a special kind of MCAP, where positions in the system can accept up to two types of components. The BITA-ZK heuristic can be divided into two stages. In the first stage, a potential assignment is initialized based on the BITA. In the second stage, local search method is applied based on the ZKD heuristic. Simulation experiments show that the BITA-ZK heuristic is very implementable in practical application, which has generally higher accuracy than the randomization method, and dramatically outperforms both the randomization method and the enumeration method by its short computation time. It should be noted that there is no other heuristics for us to conduct further comparative analyses at the moment, since this special kind of MCAP is first studied by our team.

Further research can focus on a more universal type of the MCAP, where positions in the system can accept up to more than two types of components. Also, as a BI-based local search method, the BITA-ZK heuristic inevitably converges to a local optimal assignment. Genetic algorithms may be applied to break local optimum.

Acknowledgments
This work is supported by National Natural Science Foundation of China (Grant No. 51805326 and 71632008) and National Science and Technology Major Project (Grant No. 2017-1-0007-0008).

References
[1] Zuo M and Kuo W 1990 Design and performance analysis of consecutive-k-out-of-n structure Naval Research Logistics 37 203-30
[2] Lin F-H and Kuo W 2002 Reliability importance and invariant optimal allocation Journal of Heuristics 8 155-71
[3] Yao Q, Zhu X and Kuo W 2011 Heuristics for component assignment problems based on the Birnbaum importance IIE Transactions 43 633-46
[4] Yao Q, Zhu X and Kuo W 2014 A Birnbaum-importance based genetic local search algorithm for component assignment problems Annals of Operations Research 212 185-200
[5] Cai Z, Si S, Sun S and Li C 2016 Optimization of linear consecutive-k-out-of-n system with a Birnbaum importance-based genetic algorithm Reliability Engineering & System Safety 152 248-58
[6] Zhu X, Fu Y, Yuan T and Wu X 2017 Birnbaum importance based heuristics for multi-type component assignment problems Reliability Engineering & System Safety 165 209-21