6-2018

Instance-specific selection of AOS methods for solving combinatorial optimisation problems via neural networks

Teck Hou (DENG Dehao) TENG
Singapore Management University, thteng@smu.edu.sg

Hoong Chuin LAU
Singapore Management University, hclau@smu.edu.sg

Aldy GUNAWAN
Singapore Management University, aldygunawan@smu.edu.sg

DOI: https://doi.org/10.1007/978-3-030-05348-2_9

Instance-Specific Selection of AOS Methods for Solving Combinatorial Optimisation Problems via Neural Networks

Teck-Hou Teng[2], Hoong Chuin Lau[1], Aldy Gunawan[2]

[1] School of Information Systems, Singapore Management University, Singapore
hclau@smu.edu.sg
[2] Fujitsu-SMU Urban Computing and Engineering Corporate Laboratory, Singapore Management University, Singapore
{thteng,aldygunawan}@smu.edu.sg

Abstract. Solving combinatorial optimization problems using a fixed set of operators has been known to produce poor quality solutions. Thus, adaptive operator selection (AOS) methods have been proposed. But, despite such effort, challenges such as the choice of suitable AOS method and configuring it correctly for given specific problem instances remain. To overcome these challenges, this work proposes a novel approach known as I-AOS-DOE to perform Instance-specific selection of AOS methods prior to evolutionary search. Furthermore, to configure the AOS methods for the respective problem instances, we apply a Design of Experiment (DOE) technique to determine promising regions of parameter values and to pick the best parameter values from those regions. Our main contribution lies in the use a self-organizing neural network as the offline-trained AOS selection mechanism. This work trains a variant of FALCON known as FL-FALCON using performance data of applying AOS methods on training instances. The performance data comprises derived fitness landscape features, choices of AOS methods and feedback signals. The hypothesis is that a trained FL-FALCON is capable of selecting suitable AOS methods for unknown problem instances. Experiments are conducted to test this hypothesis and compare I-AOS-DOE with existing approaches. Experiment results reveal that I-AOS-DOE can indeed yield the best performance outcome for a sample set of quadratic assignment problem (QAP) instances.

1 Introduction

Evolutionary search is commonly used to solve combinatorial optimization problems. During evolutionary search, adaptive operator selection (AOS) is used to select operators adaptively that will hopefully lead the search toward optimality. Though it may be highly probable for seasoned practitioners to gain specialized knowledge of performance characteristics for a subset of problem instances, it is a non-trivial endeavor to possess such specialized knowledge for all problem instances. Moreover, with the availability of more sophisticated machine learning techniques, such specialized knowledge may have become obsolete.

Following on a work that investigates varying operator settings [18], there has been interest on designing better AOS mechanism. Evidences of such interest

include improving bandit-based operator selection mechanisms by [4, 5]. There is also a study on the use of adaptive pursuit in neuro-evolution algorithm [8]. These works contributed to the state-of-the-art by extending specific AOS methods. To our knowledge, there has been no attempt to achieve a generic method for selecting AOS methods at the problem instance level. Also of interest to this work is on the topic of fitness landscape analysis [12], the use of fitness landscape for selecting differential evolution operators [13] and algorithm selection problem [2]. This work is also encouraged by a neural network-based AOS method using fitness landscape features for selecting crossover operators [16].

This work aims to overcome the problem of selecting suitable AOS methods and configuring it to parameter values correct for specific problem instances [18]. Thus, we propose a novel approach, known as I-AOS-DOE, integrating self-organizing neural networks and design of experiment (DOE) techniques for instance-specific selection of AOS methods. Using DOE [6], our proposed approach treats the parameter values and AOS operators as decision variables. DOE was initially applied to identify the important parameters and determine ranges of their values. Further analysis is performed to identify the best parameter values in those ranges. A class of self-organizing neural networks known as Fusion Architecture for Learning and Cognition (FALCON) [14] is used as an adaptive selector of AOS methods. A specific variant of FALCON known as FL-FALCON [17] is trained using performance data of AOS methods on the problem instances. The performance data comprises derived landscape features, the selected AOS, the AOS parameter set and feedback signals indicating the solution quality. The hypothesis of this work is that the trained FL-FALCON can select suitable AOS methods for unknown problem instances. To test this hypothesis, experiments were conducted to evaluate the performance of the proposed approach using Quadratic Assignment Problem (QAP) instances. Comparing the performance of I-AOS-DOE with existing approaches, experiment results reveal that I-AOS-DOE can indeed yield the best performance outcome for a sample set of QAP instances.

The presentation of this work continues in Section 2 where several related works are reviewed. The problem addressed through this work is defined in Section 3. The proposed approach is presented in Section 4. This is followed by the presentation of the experiments and the results in Section 5. Last but not least, Section 6 summarizes and conclude this work.

2 Related works

SATzilla was introduced as an algorithm portfolio selection methodology to solve SAT problem [20]. This is achieved by using a number of problem-specific features for a given SAT instance. The goal is to provide a runtime prediction model for SAT solvers. Later, SATzilla was enhanced to include explicit cost-sensitive loss function [21]. Earlier, Hydra was proposed as using parameter tuning to address algorithm portfolios [19]. Hydra tunes the solver using ParamILS and add parameterization to a SATzilla portfolio. In addition, ISAC was introduced as an instance-specific algorithm configurator for solving MaxSAT problems [7]. It overcomes the problem of proper parameterizing of algorithms by clustering

training instances produced by those algorithms on MaxSAT problems. Later, ISAC++ generalizes tuning of individual solvers and combine multiple solvers into a solver portfolio [1]. ISAC++ takes an additional step of using any algorithm selector to choose one of the parameterizations.

SNAP-NEAT incorporates adaptive pursuit as the AOS mechanism for solving fractured problems [8]. To do so, SNAP-NEAT makes continuous updates and initial estimation of the operator values and probability. Evaluated using several problem instances, the experiment results reveal that SNAP-NEAT can select the best operators intelligently for the problem instances. Thus, SNAP-NEAT has demonstrated the ability to combine the strengths of NEAT, RBF-NEAT and Cascade-NEAT effect for solving reactive control and high-level problems.

A dynamic variant of the Multi-Armed Bandit (MAB) upper confidence bound algorithm was introduced as an AOS mechanism [5]. It uses a sliding window to update operator quality estimates, discard ancient events while preserving the recent ones. The MAB-based AOS mechanism was evaluated using artificial and real optimization problems for operator quality distribution. There is also an MAB-based AOS mechanism using fitness landscape analysis to better describe the resultant population during evolutionary search [4]. An online learning algorithm known as dynamic weighted majority (DWM) is used to model concept drifts. It was evaluated using several problem instances and compared with MAEN*-II. Other AOS mechanisms using fitness landscape analysis includes a landscape-based AOS mechanism for differential evolution (LSAOS-DE) for selecting multiple DE operators [13]. LSAOS-DE uses problem landscape information and performance histories of operators in an adaptive operator selection mechanism. LSAOS-DE is evaluated using single-objective optimization functions from CEC2014 and CEC2015 competitions. It is also compared with other heuristic-based selection method and state-of-the-art algorithms.

One-sided support vector regression was introduced to address the algorithm selection problem [2]. It employs exploratory landscape analysis to discover landscape features. That proposed approach is evaluated using several BBOB functions. The BBOB functions are divided into four classes according to modality, separability and global structure. Results from the experiments shows that one-sided support vector regression can generalize better than the selected benchmarks. It has also provided better insights on how landscape features can be mapped to efficient algorithms. There is also a meta-learning framework for addressing algorithm selection problem in continuous optimization problems [10]. The independent variables are the landscape features and algorithm parameters while the dependent variable is the algorithm performance. Performance of the proposed neural network-based approach is evaluated using the CEC2005 benchmarks. The predicted rankings made by the proposed approach are compared with the actual rankings and random ranking.

3 Problem Statement

This work addresses the problem of solving quadratic assignment problem (QAP) instances unknown to the solver. Landscape features are derived using fitness values from a different and unknown solver. Only the derived landscape features are

presented as input to the solver for solving QAP problem instances. The generated solution is evaluated using fitness function on the QAP problem instances. In this work, there are several QAP instances, AOS methods and possible values for the AOS parameters. The performance of the AOS methods across the problem instances can be rather different [20]. Thus, the challenge here is to perform instance-specific selection of AOS methods for specific problem instances.

The application of solver to QAP problem instance j results in a fitness landscape comprising a list of objective values. A QAP problem instance has also an objective value of a best-known solution. This work subtracts the objective value of best-known solution from the objective values in the fitness landscape to obtain a list of objective value deviations. A set of fitness landscape features $\mathcal{F}^j$ for problem instance j is derived using the objective value deviations. The proposed solution is to recommend the choice of AOS method aos_i and possible values for AOS parameters ap_i^h for solving unknown problem instance j. Clearly, the resultant fitness landscape and solution is expected to be better than the quality of the initial fitness landscape $\mathcal{F}^j$. The quality of the fitness landscape is quantified using mean value and minimum value of the objective value deviation.

4 Instance-Specific Selection of AOS Methods

This paper proposed a framework, illustrated using Fig. 1, to address the problem of solving combinatorial optimization problems. Known as I-AOS-DOE, it performs instance-specific selection of AOS methods prior to conducting evolutionary search. I-AOS-DOE uses an artificial neural network to make good recommendations of AOS methods and parameter values for problem instances adaptively. The statistical technique known as Design of Experiment (DOE) is used to identify good parameter values for the AOS methods.

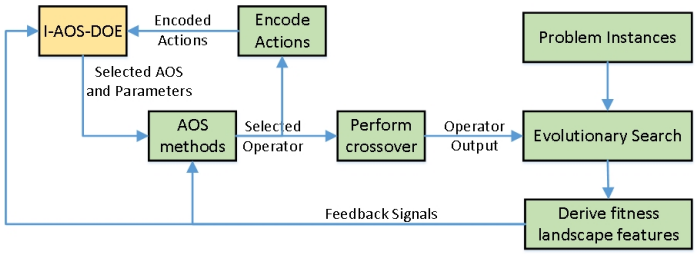

Fig. 1. Illustration of the proposed I-AOS-DOE framework for selecting AOS methods to unknown problem instances

4.1 Neural Network-based Approach

I-AOS-DOE uses a class of ART-based self-organizing neural network known as FALCON [14] for selecting AOS methods. FALCON is favored for the ability to learn incrementally in real time. Based on the adaptive resonance theory (ART) [3], FALCON has been demonstrated in several prior works [14–17] to be capable of striking a delicate balance between specificity and generalizability. A specific variant of FALCON known as FL-FALCON [17] is used in this work.

Structure and Operating Modes Seen in Fig. 2, FL-FALCON has a two-layer architecture, comprising an input/output (IO) layer and a knowledge layer. The IO layer has a sensory field F_1^{c1} for accepting state vector $\mathbf{S}$, an action field F_1^{c2} for accepting action vector $\mathbf{A}$, and a reward field F_1^{c3} for accepting reward vector $\mathbf{R}$. The category field F_2^c at the knowledge layer stores the committed and uncommitted cognitive nodes. Each cognitive node j has template weights $\mathbf{w}^{\mathrm{ck}}$ for $k = \{1, 2, 3\}$.

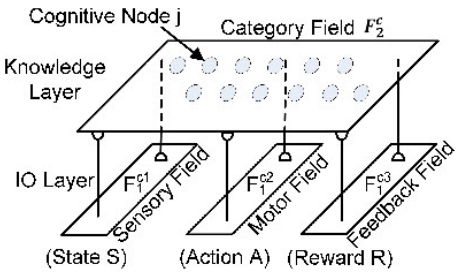

Fig. 2. The FALCON architecture.

FL-FALCON operates in one of the following operating modes. In *PERFORM* mode, FL-FALCON selects cognitive node J for deriving action choice a for state s. In *LEARN* mode, FL-FALCON learns the effect of action choice a on state s. In *INSERT* mode, domain or external knowledge can be assimilated into FL-FALCON [15].

Insertion of Performance Data This work prepares FL-FALCON for selecting AOS methods by inserting performance data of AOS methods on problem instances into it. This step prepares FL-FALCON for selecting AOS methods. Performance characteristics of AOS methods on problem instances are inserted into FL-FALCON in two stages. The first stage is to encode input pattern $\mathbf{x}^{ck}$ where derived fitness landscape features are encoded as state vector $\mathbf{x}^{c1}$, decision variables are encoded as action vector $\mathbf{x}^{c2}$ and feedback signal r are encoded as reward vector $\mathbf{x}^{c3}$, i.e., $\mathbf{x}^{ck} = \{\mathbf{x}^{c1}, \mathbf{x}^{c2}, \mathbf{x}^{c3}\}$. The second stage is to insert the encoded performance data into FL-FALCON. The insertion operation is performed by selecting a winning cognitive node J for input pattern $\mathbf{x}^{ck}$ using the code selection steps. Template learning is applied on winning cognitive node J to learn input pattern $\mathbf{x}^{ck}$.

The decision variables encoded in action vector $\mathbf{x}^{c2}$ are the identity of problem instance, the choice of AOS method and parameters of the selected AOS method. Different AOS methods have different number of parameters. To account for such differences, action vector $\mathbf{x}^{c2}$ is defined to be capable of holding the largest number of parameters. The continuous values of AOS parameters are between 0.0 and 1.0. A parameter having value x is encoded into action vector $\mathbf{x}^{c2}$ as $\{x, 1.0 - x\}$. Identity of problem instance and AOS method are nominal decision variables. To encode such information into action vector $\mathbf{x}^{c2}$, a numerical index v_i is used for decision variable i. After that, it is divided by available number of choices D_i to give a continuous numerical value d_i between 0.0 and 1.0. After that, numerical equivalent of decision variable i d_i is encoded into $\mathbf{x}^{c2}$

as $\{d_i, 1.0 - d_i\}$. This way of encoding a numerical value using two elements is known as complement coding.

Code Selection Code selection is performed to identify winning cognitive node J for deriving action choice a for state s in *PERFORM* mode. A winning cognitive node J is identified in *LEARN* mode to learn the mapping of state s to action choice a with effect r. In *INSERT* mode, a winning cognitive node J is identified to learn an unit of domain knowledge.

A winning cognitive node J is identified in two stages. The first stage is known as code activation. This stage derive choice function T_j^c for the committed cognitive node j with respect to input pattern $\mathbf{x}^{ck}$ for $k = \{1, 2, 3\}$ using

$$T_j^c = \sum_{k=1}^{3} \gamma^{ck} \frac{|\mathbf{x}^{ck} \wedge \mathbf{w}_j^{ck}|}{\alpha^{ck} + |\mathbf{w}_j^{ck}|} \tag{1}$$

where the fuzzy AND operation $(\mathbf{p} \wedge \mathbf{q})_i \equiv min(p_i, q_i)$, the norm $|.|$ is defined by $|\mathbf{p}| \equiv \sum_i p_i$ for vectors $\mathbf{p}$ and $\mathbf{q}$, $\alpha^{ck} \in [0, 1]$ is the choice parameters, $\gamma^{ck} \in [0, 1]$ is the contribution parameters and $k = \{1, 2, 3\}$. This is followed by a competition among the cognitive nodes using their choice function T_j^c to find winning cognitive node J using

$$J = \arg\max_j \{T_j^c : \text{for all } F_2^c \text{ node } j\} \tag{2}$$

This is followed by the second stage where the match functions m_J^{ck} of winning cognitive node J with respect to input pattern $\mathbf{x}^{ck}$ is derived and checked against vigilance parameters ρ^{ck} using

$$m_J^{ck} = \frac{|\mathbf{x}^{\mathrm{ck}} \wedge \mathbf{w}_J^{ck}|}{|\mathbf{x}^{\mathrm{ck}}|} \geq \rho^{ck} \tag{3}$$

A winning cognitive node J is only confirmed when it satisfies the vigilance criterion as expressed using (3). Using winning cognitive node J, FL-FALCON performs operation specific to the operating mode. In *LEARN* and *INSERT* modes, FL-FALCON executes the template learning operation. In *PERFORM* mode, FL-FALCON executes the activity readout operation using $\mathbf{w}_J^{c2}$.

Template Learning This step is executed to learn an input pattern $\mathbf{x}^{ck}$ using a winning cognitive node J identified using code selection. Winning cognitive node J can either be a committed or uncommitted cognitive node. A committed cognitive node j contains a previously learned pattern while an uncommitted cognitive node j has not learned an input pattern.

Template learning takes place by modifying weight $\mathbf{w}_J^{ck}$ of winning cognitive node J using

$$\mathbf{w}_J^{ck(\mathrm{new})} = (1 - \beta^{ck})\mathbf{w}_J^{ck(\mathrm{old})} + \beta^{ck}(\mathbf{x}^{ck} \wedge \mathbf{w}_J^{ck(\mathrm{old})}) \tag{4}$$

where $\beta^{ck} \in [0, 1]$ is the learning rate. After that, cognitive node J becomes a part of the knowledge layer. It will participate in subsequent rounds of code selection.

Activity Readout This is the process of acquiring action choice a from action field $\mathbf{w}_J^{c2}$ of winning cognitive node J. Previous works encode a single decision variable in $\mathbf{w}_J^{c2}$. By doing so, action choice a can be read out using

$$\mathbf{x}^{c2(new)} = \mathbf{x}^{c2(old)} \wedge \mathbf{w}_J^{c2} \tag{5}$$

In this work, decision variables are encoded as action vector $\mathbf{x}^{c2}$. The readout of the decision variables from action field $\mathbf{w}_J^{c2}$ of winning cognitive node J is performed in reverse order to the encoding scheme.

4.2 Input Pattern

Input pattern $\mathbf{x}^{ck}$ to FL-FALCON comprises state vector $\mathbf{x}^{c1}$, action vector$\mathbf{x}^{c2}$ and reward vector $\mathbf{x}^{c3}$. State vector $\mathbf{x}^{c1}$ encodes the fitness landscape features. Action vector $\mathbf{x}^{c2}$ encodes the decision variables. Reward vector $\mathbf{x}^{c3}$ encodes the feedback signal.

Fitness Landscape Features Combinatorial optimization problem instances are solved using evolutionary search. Solution s_i^j generated at search iteration i for problem instance j has an objective value $o(s_i^j)$. Problem instance j has an optimal solution os^j with objective value $o(os^j)$. Objective value deviation δ_i^j of solution s_i^j at search iteration i to optimal solution os^j is derived using

$$\delta_i^j = \frac{|o(s_i^j) - o(os^j)|}{o(os^j)}. \tag{6}$$

Evolutionary search is conducted for $\mathcal{N}$ search iterations. Fitness landscape $\mathcal{F}^j$ is formed using $\mathcal{N}$ values of objective value deviation. Fitness landscape analysis (FLA) [12] and descriptive statistics techniques are applied on fitness landscape $\mathcal{F}^j$ to derive 10 landscape-based features comprising the minimum value (min), the maximum value (max), the median (med), the standard deviation ($std-dev$), the coefficient of variation (cv), range (r), skewness (sk), kurtosis (kt), proportion of upticks ($uptick$) and proportion of downticks ($downticks$).

For minimizing optimization problem j, min feature represents an objective value deviation δ_i^j of solution s_i^j having an objective value $o(s_i^j)$ closest to objective value $o(os^j)$ of optimal solution os^j. Similarly, max feature represents an objective value deviation δ_k^j of solution s_k^j having an objective value $o(s_i^j)$ furthest away from objective value $o(os^j)$ of optimal solution os^j. The difference between $o(s_i^j)$ and $o(s_k^j)$ gives the range of objective value deviation of fitness landscape $\mathcal{F}^j$. The standard deviation $std - dev$ feature measures the average amount of quantitative deviation from the mean value of fitness landscape $\mathcal{F}^j$. The coefficient of variation cv measures the dispersion of the objective value deviation δ_i^j making up fitness landscape $\mathcal{F}^j$. For this work, skewness (sk) is a measure of symmetry of fitness landscape $\mathcal{F}^j$. Kurtosis (kt) is a measure of *peakedness* of fitness landscape $\mathcal{F}^j$. With respect to search i, an uptick represents a worse solution is found at search iteration $i+1$. The $uptick$ feature is the proportion of upticks with respect to $\mathcal{N}$ search iterations. Similarly, a downtick

represents a better solution is found at search $i+1$. The *downtick* feature is the proportion of downticks with respect to $\mathcal{N}$ search iterations. The derived landscape features are normalized and encoded using complement coding.

Decision Variables The decision variables comprises the problem instance, the AOS method and values of AOS parameters. The choices of problem instance and AOS method have nominal values while the parameters are normalized continuous values. Thus, the problem instance and AOS method choices are encoded as part of the action vector by representing them numerically as indices. The problem index and AOS index are then normalized using the total number of problem instances and AOS methods respectively. The AOS methods have different number of tunable parameters. Thus, a fixed-length action vector is defined to accommodate the largest number of AOS parameters. Similar to the state vector, complement coding is used to encode the decision variables into the action vector. A compatible decoding scheme is used to retrieve the decision variables from the action vector.

Feedback Signal This is a continuous value indicating the *quality* of fitness landscape for problem instance j. The quality of fitness landscape is measured with respect to the objective value of its optimal solution. Feedback signal f^j is derived as an average of the mean value $m(\delta^j)$ and the minimum value $min(\delta_i^j)$ of objective value deviations. The mean value gives an aggregated perception of *proximity* of raw fitness values to the fitness value of the optimal solution for a problem instance. Thus, smaller mean value indicates fitness landscape with better quality. The minimum value of objective value deviation represents the best possible solution for a problem instance. Thus, smaller minimum value of objective value deviation indicates the presence of better quality solution. Due to such significance, the feedback signal f^j is derived using $\frac{m(\delta^j)+min(\delta_i^j)}{2}$

4.3 Select AOS Methods

Properly parameterized AOS methods are selected in two stages. The first stage is to insert performance data into FL-FALCON. This is following by the second stage of selecting AOS methods using the trained FL-FALCON. The steps for inserting performance data into FL-FALCON is outlined using Algorithm 1. It entails the derivation of fitness landscape features $\mathcal{F}^j$ for problem instance j. The derived fitness landscape features $\mathcal{F}^j$ are then encoded as state vector $\mathbf{x}^{c1}$. The decision variables and feedback signals are encoded as action vector $\mathbf{x}^{c2}$ and reward vector $\mathbf{x}^{c3}$ respectively. Together, they form the input pattern $\mathbf{x}^{ck}$ to be inserted into FL-FALCON.

The trained FL-FALCON selects the AOS methods along with parameter values using the steps outlined in Algorithm 2. The proposed approach has two loops. Given the derived fitness landscape features $\mathcal{F}$, the inner loop search for the best response from FL-FALCON. The winning cognitive node J is inhibited after each iteration of the inner loop to allow other cognitive nodes to win. The chosen response comprising AOS method and parameter values are returned to

Algorithm 1 Insertion of Performance Data into FL-FALCON

Require: performance data for applying AOS methods on problem instances
Ensure: FL-FALCON is initialize to appropriate structure and parameter values.
while has more performance data **do**
  derive fitness landscape features from raw fitness landscape
  encode derived fitness landscape features $\mathcal{F}^j$ as state vector $\mathbf{x}^{c1}$
  encode decision variables as action vector $\mathbf{x}^{c2}$
  encode feedback signal as reward vector $\mathbf{x}^{c3}$
  insert input pattern $\mathbf{x}^{ck} = \{\mathbf{x}^{c1}, \mathbf{x}^{c2}, \mathbf{x}^{c3}\}$ into FL-FALCON operating in *INSERT* mode
end while
return trained FL-FALCON

the outer loop where it is used in evolutionary search. After that, a new set of derived fitness landscape features are obtained using the collected objective values. Together with the chosen response and feedback signals, it is presented to FL-FALCON for further learning. The updated FL-FALCON is used in subsequent rounds of search for better AOS methods and parameter sets. The outer loop terminates after meeting the closing criteria.

Algorithm 2 Solving combinatorial optimization problem using I-AOS-DOE

Require: trained FL-FALCON
Require: derived fitness landscape features $\mathcal{F}$
Require: State vigilance threshold ζ^{c1} and state vigilance stepsize $\delta\zeta^{c1}$
Ensure: best reward $r* = 0.0$
while not end of search **do**
  encode derived fitness landscape features $\mathcal{F}$ as state vector $\mathbf{x}^{c1}$
  set action vector $\mathbf{x}^{c2}$ and reward vector $\mathbf{x}^{c3}$ to uncommitted patterns
  while $\rho_p^{c1} > \zeta^{c1}$ **do**
    present input pattern $\mathbf{x}^{ck}$ to FL-FALCON for action selection
    if $w_J^{c3}(0) > r*$ **then**
      $r* = w_J^{c3}(0)$
      readout AOS methods and parameter values from action vector $\mathbf{w}_J^{c2}$
    end if
    inhibit winning cognitive node J
    reduce state vigilance ρ_p^{c1} using $\delta\zeta$
  end while
  clear inhibition for all cognitive nodes
  if no valid response **then**
    select AOS method and parameter values randomly
  end if
  apply AOS method configured with DOE-derived parameter values to solve problem instance j
  remember best solution to problem instance j
  derive fitness landscape features $\mathcal{F}^j$ from a new set of objective values
  encode derived fitness landscape features $\mathcal{F}$ as state vector $\mathbf{x}^{c1}$
  encode selected response as action vector $\mathbf{x}^{c2}$
  encode feedback signal as reward vector $\mathbf{x}^{c3}$
  present input pattern $\mathbf{x}^{ck} = \{\mathbf{x}^{c1}, \mathbf{x}^{c2}, \mathbf{x}^{c3}\}$ to FL-FALCON for learning
end while
return best solution to problem instance j

4.4 Design of Experiment

Design of Experiment (DOE) is a well-studied statistical technique for determining key parameters of particular process [9] and best parameter values for target

algorithms [6]. In this paper, we focus on defining promising regions for the parameter configurations and define the best parameter values for AOS methods using DOE technique. We briefly explain the idea of this approach.

Given a set of parameters K, it is assumed that the initial range value of each parameter $k \in K$ is known and bounded by a numerical interval $[LB_k, UB_k]$. A $2^{|K|}$ factorial design is applied to screen and rank the parameters. A complete design requires $rep \times 2^{|K|}$ observations where rep represents the number of replicates for a set of parameter values. Screening is performed to determine the parameters that are statistically significant. This step reduces the number of parameters to tune. The following example provides an illustration of how the screening phase is applied.

Suppose we wish to study the effect of two parameters k_1 and k_2. The 2^2 factorial design would consist of four experimental units where each unit is run rep times:

unit (1): set k_1 at LB_{k_1} and k_2 at LB_{k_2} unit (k_1): set k_1 at LB_{k_1} and k_2 at UB_{k_2}
unit (k_2): set k_1 at UB_{k_1} and k_2 at LB_{k_2} unit (k_1k_2): set k_1 at UB_{k_1} and k_2 at UB_{k_2}

A factorial experiment is then analyzed using Analysis of Variance (ANOVA) to estimate the main effect for a particular parameter. The test of significance of the main effect of the parameters with a significance level (e.g. $\alpha = 5\%$) is conducted for determining the importance of parameters. The ranking of the important parameters is then done by comparing the absolute values of the main effects of those parameters. Each non-significant parameter is then set to a constant value. After that, we find promising ranges for m important parameters for $m \leq |K|$. This process begins by exploring a larger space where the linear relationship holds allowing for the application of standard approach for linear model checking and diagnosis [9]. The AOS is run with respect to the parameter configuration space which contains $(2^m + 1)$ possible parameter configurations with an additional setting defined by the centre points of the m important parameters. Centre points are added to protect against curvature.

The interaction and curvature tests are used for checking model adequacy. The planar model can still be applied as long as either one of them is not statistically significant. Otherwise, the region of planar local optimality has been reached and the promising region has been found. The process is continued by applying the steepest descent, in order to bring the parameter to the vicinity of the optimum values. Once the region of the optimum values has been found (e.g. one of two statistical tests is statistically significant), the planar model becomes invalid. The important parameters are assumed to be in their promising range. The best parameter value is taken by discretizing the range and pick the value that provides the best result. Details on the DOE technique are in [6].

5 Performance Evaluation

Experiments are conducted to evaluate the performance of I-AOS-DOE and compared it with selected AOS methods and crossover operators. To do so, it is necessary to inform the readers of the necessary details of the experiments such as the choice of AOS methods and crossover operators in Section 5.1, the default parameters and DOE-derived parameters in Section 5.2 and the selected

QAP problem instances in Section 5.3. After that, the results are presented and analysed in Section 5.4.

5.1 AOS and Crossover Operators

The AOS methods and crossover operators used in this work are presented in the following paragraphs.

Adaptive Operator Selection As mentioned in [4], adaptive operator selection is composed of credit assignment and operator selection sub-tasks. The credit assignment sub-task allocates a credit value signifying the effect of the selected operator on the quality of the solution. The operator selection sub-task decides on the choice of operators based on the knowledge of credit value for the operators. Several AOS methods implementing different approaches for the sub-tasks are known. In this work, the AOS methods that can be selected using I-AOS-DOE are probabilistic matching (PM), adaptive pursuit (AP), multi-armed bandit (MAB), reinforcement learning (RL) and a self-organizing artificial neural network (NN) known as FL-FALCON [17].

Crossover Operators The AOS methods are implemented to select crossover operators during the search iterations. Crossover operator picks parts of the parent chromosomes and use it to form a child chromosome. In this way, the child chromosome may possess characteristics of the parent chromosomes. Different crossover operators implement the crossover operation differently. The crossover operators used in this work are the cycle crossover (CX), distance-preserving crossover (DPX), partially-mapped crossover (PMX) and order crossover (OX). Details on the crossover operators are available in [16].

5.2 Design of Experiment

The steps of DOE are presented in Section 4.4. Table 1 summarizes the initial ranges for parameters of each AOS method. One column is used to present the best parameter values for each AOS when solving the training instances. For example, the best parameters for Reinforcement Learning are $\alpha = 0.2$ and $\delta = 1.0$. The last column contains the default parameters.

Table 1. Paramater values of AOS

AOS	par	$[LB_{par}, UB_{par}]$	Best Value	Default Value [16]
Probability Matching	α	$[0.0, 1.0]$	0.5	0.3
	p_{min}	$[0.0, 0.2]$	0.1	0.05
Adaptive Pursuit	α	$[0.0, 1.0]$	0.5	0.3
	β	$[0.0, 1.0]$	1.0	0.3
	p_{min}	$[0.0, 0.2]$	0.2	0.05
Multi-armed Bandit	γ	$[0.0, 1.0]$	0.5	1.0
Reinforcement Learning	α	$[0.0, 0.2]$	0.2	0.03
	δ	$[0.0, 1.0]$	1.0	0.9
Neural Network	ρ_p^{c1}	$[0.9, 1.0]$	1.0	0.97
	ρ_p^{c2}	$[0.9, 1.0]$	0.95	0.0
	ρ_p^{c3}	$[0.0, 0.1]$	0.05	0.03

5.3 Problem Instances

This work focuses on solving selected Quadratic Assignment Problem (QAP) problem instances for demonstrating the efficacy of I-AOS-DOE. The QAP problem instances are selected using prior knowledge of the difficulty levels. The primary source of such knowledge is the work on the use of self-organizing neural network as an AOS method in [16]. It is discovered that the AOS methods and crossover operators have rather broad range of performance profile on 18 QAP instances. Thus, the interest here is to propose a robust approach capable of stabilizing the performance on the following 18 QAP problem instances.

chr15a, chr15c, chr18a, chr20a, chr20b, chr20c, chr22a, chr22b, chr25a, kra30a, kra30b, kra32, lipa40a, nug17, nug24, nug28, nug30, rou20,

5.4 Result Analysis

The results used to illustrate the performance of the AOS methods and crossover operators are seen in Fig. 3 and Fig 4. The results are presented in the form of ranks among the various approaches. These approaches are ranked according to their best objective values for solving the selected QAP problem instances.

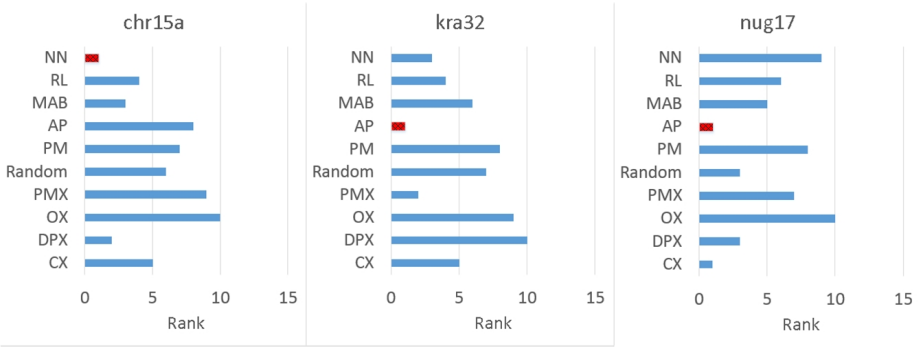

Fig. 3. Illustration of the differences in performance of AOS methods on different problem instances

It is known from [16] that there are problem instances where the AOS methods have rather diverse outcome. Fig. 3 is meant to illustrate this observation. From Fig. 3, NN is observed having the best performance for chr15a problem instance whereas AP is observed performing better than NN for kra32 and nug17 problem instances. In addition, the other AOS methods have rather different performance outcome among these problem instances as well. Thus, I-AOS-DOE is designed to exploit the strength of AOS methods for different problem instances.

From Fig. 4, I-AOS-DOE is showing rather robust and stable performance. It has the best rank among the approaches in both experiments. At the other end of the spectrum, order crossover operator (OX) has the worst performance aggregated over the selected problem instances. The worst performing AOS method configured using the default parameters is AP. The worst performance AOS method configured using DOE-derived parameters is RL.

In the second experiment, only the parameters of the AOS methods are derived using DOE. Thus, the effect of DOE on the performance of AOS methods

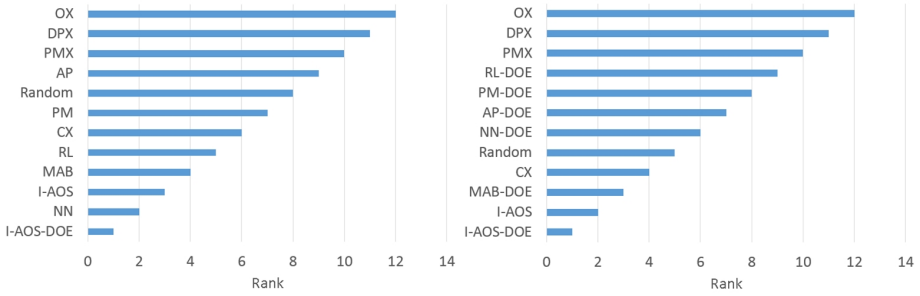

Fig. 4. Illustration of the ranking obtained by AOS methods configured using DOE-derived parameters (left) and default parameters (right) and the crossover operators.

can be observed objectively using Fig. 5. The objective value difference seen in Fig. 5 is obtained by subtracting the objective value of the approach whose parameter values are derived using DOE from that of the same approach configured with the default parameter values. This approach is taken because of the expectation that DOE-derived parameters can help the AOS methods to improve on their performance. Thus, if the application of DOE has helped, the objective value difference will be positive and vice versa. The approaches seen in Fig. 5 are ranked in ascending order with respect to their objective value difference. Positive objective value difference indicates the application of DOE has helped in improving the performance of the AOS method for solving combinatorial optimisation problem using evolutionary search.

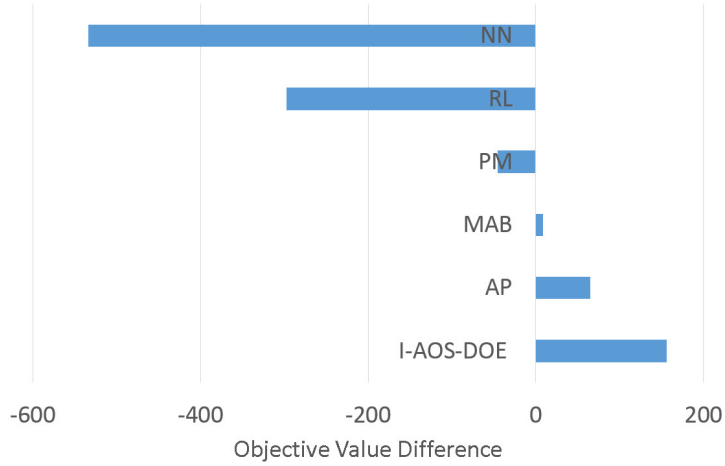

Fig. 5. Illustrations of the ranking obtained on the effect of DOE on the performance of the AOS methods

I-AOS-DOE is only affected by the application of DOE indirectly. This is because the parameters of I-AOS-DOE are not tuned using DOE. Rather, it is the parameters of AOS methods selected by I-AOS-DOE that are derived using DOE. From Fig. 5, I-AOS-DOE, MAB and AP are observed having positive objective value differences. This means the performance of these three approaches are helped by the application of DOE. Given that the parameters of I-AOS-DOE

are not tuned using DOE, its performance improvement can be attributed to the selection of AOS methods helped by the application of DOE.

6 Conclusions

This work proposed a neural network-based approach for performing instance-specific selection of adaptive operator selection methods known as I-AOS-DOE. Using a class of self-organizing neural network known as FALCON, I-AOS-DOE is trained to select AOS methods for solving specific QAP problem instances. Training of a specific variant of FALCON known as FL-FALCON is carried out by inserting performance data of AOS methods on QAP problem instances. The performance data comprises a set of derived fitness landscape features as the state features, a set of decision variables comprising identity of problem instance, the choice of AOS method and the set of AOS parameter values as features of the action space. A feedback signal aggregating the average objective value difference and the best objective value difference to the objective value of a known optimal solution is encoded as the reward vector.

The trained FL-FALCON selects a set of decision variables using a set of derived fitness landscape features as inputs. Without knowing the identity of the problem instance, the trained FL-FALCON is to recommend a AOS method along with the appropriate parameter set for solving the problem instance using evolutionary search. Experiments were conducted using 18 QAP problem instances, 5 AOS methods and 4 crossover operators. Experiment results are obtained for AOS methods configured using the default parameter values and DOE-derived parameter values. The aggregated effect of these two sets of AOS methods over the selected QAP problem instances are compared and contrasted. From the results, it is observed that I-AOS-DOE is most effective and robust in solving the selected QAP instances. In addition, it is shown that DOE is effective towards certain AOS methods in solving the selected QAP instances using evolutionary search. Consequentially, the use of DOE turns out to have positive effect on the performance of I-AOS-DOE.

This work can be extended in several dimensions. In particular, we might incorporate I-AOS-DOE into the Hyflex framework [11]. By being in this Hyflex framework, I-AOS-DOE can be evaluated on problem domains such as maximum satisfiability problem (SAT), the flow shop sequencing problem, the bin packing problem and the personnel scheduling. It may also be possible to compare against other approaches for solving these benchmark problems. Beyond that, there are also many fitness landscape analysis (FLA) techniques that can be exploited to better analyse and characterise the raw fitness landscape of problem instances. The derivation of fitness landscape features agnostic to approaches used for solving problem instances can help I-AOS-DOE learn a set of generalizable policies for selecting AOS methods for unknown problem instances.

Acknowledgments

This research project is funded by National Research Foundation Singapore under its Corp Lab @ University scheme and Fujitsu Limited.

Bibliography

[1] Carlos Ansotegui, Joel Gabas, Yuri Malitsky, and Meinolf Sellmann. MaxSAT by improved instance-specific algorithm configuration. *Artificial Intelligence*, 235:26–39, 2016.

[2] Bernd Bischl, Olaf Mersmann, Heike Trautmann, and Mike Preuß. Algorithm selection based on exploratory landscape analysis and cost-sensitive learning. In *Proceedings of the 14th annual conference on Genetic and evolutionary computation*, pages 313–320. ACM, 2012.

[3] G. A. Carpenter and S. Grossberg. A massively parallel architecture for a self-organizing neural pattern recognition machine. *Computer Vision, Graphics, and Image Processing*, 37(1):54115, January 1987.

[4] Pietro A Consoli, Yi Mei, Leandro L Minku, and Xin Yao. Dynamic selection of evolutionary operators based on online learning and fitness landscape analysis. *Soft Computing*, 20(10):3889–3914, 2016.

[5] Álvaro Fialho, Luis Da Costa, Marc Schoenauer, and Michèle Sebag. Analyzing bandit-based adaptive operator selection mechanisms. *Annals of Mathematics and Artificial Intelligence*, 60(1):25–64, 2010.

[6] A. Gunawan, H. C. Lau, and Lindawati. Fine-tuning algorithm parameters using the design of experiments approach. In C.A. Coello Coello, editor, *LION'11*, LNCS, pages 278–292. Springer, 2011.

[7] Serdar Kadioglu, Yuri Malitsky, Meinolf Sellmann, and Kevin Tierney. ISAC–instance-specific algorithm configuration. In *Proceedings of the 2010 Conference on ECAI 2010: 19th European Conference on Artificial Intelligence*, pages 751–756, 2010.

[8] Nate Kohl and Risto Miikkulainen. An integrated neuroevolutionary approach to reactive control and high-level strategy. *IEEE Transactions on Evolutionary Computation*, 16(4):472–488, 2012.

[9] D.C. Montgomery. *Design and Analysis of Expeirments*. John Wiley and Sons Inc, 6th Edition, 2005.

[10] Mario A Muñoz, Michael Kirley, and Saman K Halgamuge. A meta-learning prediction model of algorithm performance for continuous optimization problems. In *International Conference on Parallel Problem Solving from Nature*, pages 226–235. Springer, 2012.

[11] Gabriela Ochoa, Matthew Hyde, Tim Curtois, Jose Vazquez-Rodriguez, James Walker, Michel Gendreau, Graham Kendall, Barry McCollum, Andrew Parkes, Sanja Petrovic, et al. Hyflex: A benchmark framework for cross-domain heuristic search. *Evolutionary Computation in Combinatorial Optimization*, pages 136–147, 2012.

[12] Erik Pitzer and Michael Affenzeller. A comprehensive survey on fitness landscape analysis. *Recent Advances in Intelligent Engineering Systems*, 378:161–191, 2012.

[13] Karam M Sallam, Saber M Elsayed, Ruhul A Sarker, and Daryl L Essam. Landscape-based adaptive operator selection mechanism for differential evolution. *Information Sciences*, 418:383–404, 2017.

[14] A.-H. Tan. FALCON: A Fusion Architecture for Learning, COgnition, and Navigation. In *Proceedings of the IJCNN*, pages 3297–3302, 2004.

[15] T.-H. Teng, A.-H. Tan, and J. M. Zurada. Self-organizing neural networks integrating domain knowledge and reinforcement learning. *IEEE Transactions on Neural Networks and Learning Systems*, 26(5):889–902, 2015.

[16] Teck-Hou Teng, Stephanus Daniel Handoko, and Hoong Chuin Lau. Self-organizing neural network for adaptive operator selection in evolutionary search. In *International Conference on Learning and Intelligent Optimization*, pages 187–202. Springer, 2016.

[17] Teck-Hou Teng and Ah-Hwee Tan. Fast reinforcement learning under uncertainties with self-organizing neural networks. In *Proceedings of IAT*, pages 51–58, December 2015.

[18] Andrew Tuson and Peter Ross. Adapting operator settings in genetic algorithms. *Evol. Comput.*, 6(2):161–184, June 1998.

[19] Lin Xu, Holger H. Hoos, and Kevin Leyton-Brown. Hydra: Automatically configuring algorithms for portfolio-based selection. In *Proceedings of the 24th AAAI conference on Artificial Intelligence*, pages 210–216, 2010.

[20] Lin Xu, Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. SATzilla: Portfolio-based algorithm selection for SAT. *Journal of Artificial Intelligence Research*, 32(1):565–606, 2008.

[21] Lin Xu, Frank Hutter, Jonathan Shen, Holger H Hoos, and Kevin Leyton-Brown. Satzilla2012: Improved algorithm selection based on cost-sensitive classification models. *Proceedings of SAT Challenge*, pages 57–58, 2012.