"Adaptive Crossover Operator Based Multi-objective Binary Genetic Algorithm for Feature Selection in Classification". It was published in Knowledge-Based Systems in 2021 by Yu Xue, Haokai Zhu, Jiayu Liang, and Adam Słowik.

The paper presents a novel approach to feature selection in classification problems using a multi-objective binary genetic algorithm (MOBGA) with an adaptive crossover operator selection mechanism (MOBGA-AOS). Feature selection is crucial in classification as it involves removing irrelevant or redundant features from a dataset. The authors argue that feature selection can be seen as a multi-objective optimization problem, focusing on reducing the number of features while improving classification accuracy.

Their proposed MOBGA-AOS integrates five different crossover operators with distinct search characteristics. These operators are selected adaptively during the evolutionary process based on their performance, enhancing the ability to generate optimal feature subsets. The approach aims to balance exploration and exploitation in the search process effectively.

The performance of MOBGA-AOS is compared with other evolutionary multi-objective algorithms across various datasets. The results demonstrate MOBGA-AOS's effectiveness in reducing feature count while maintaining low classification error, particularly in large-scale datasets, showcasing its suitability for high-dimensional feature selection problems.

Overall, the paper contributes to the field of feature selection in machine learning by introducing an adaptive mechanism in genetic algorithms, enhancing their applicability and efficiency in complex, high-dimensional datasets.

In the paper "Adaptive Crossover Operator Based Multi-objective Binary Genetic Algorithm for Feature Selection in Classification," fitness evaluations play a crucial role in the proposed multi-objective binary genetic algorithm (MOBGA-AOS). The fitness evaluation process is described as follows:

1. Purpose of Fitness Evaluation: The goal is to find an optimal set of features that achieve high classification accuracy with a small solution size (i.e., number of features)

2. Fitness Functions:

- First Fitness Function: Minimizing classification error, rather than maximizing classification accuracy. The classification error is calculated using a specific formula.

- Second Fitness Function: The size of the solution, which is the number of features used.

- Classifier for Evaluation: k-Nearest Neighbors (k-NN, with k = 3) is employed as the classifier to evaluate solutions, along with n-fold cross-validation (n = 3).

Overall, fitness evaluations in MOBGA-AOS are integral for determining the effectiveness of feature subsets in terms of their classification accuracy and size, and for guiding the genetic algorithm towards optimal solutions.

Crossovers:

In the paper "Adaptive Crossover Operator Based Multi-objective Binary Genetic Algorithm for Feature Selection in Classification," the authors introduce an innovative approach to crossover operations within the context of a genetic algorithm. Here's a detailed explanation of the different types of crossover operators used in their study:

1. Shuffle Crossover Operator: This operator randomly shuffles the genes of two chromosomes before applying a one-point crossover. After selecting a random crossover point, two children are generated. Subsequently, the genes of the children are unshuffled in the same manner as they were initially shuffled.

2. Single-Point Crossover Operator: A single crossover point is randomly selected on two parent chromosomes. The genes are then swapped beyond or after this point to generate two children.

3. Two-Point Crossover Operator: Similar to the single-point crossover, but here, two points are randomly selected on the parent chromosomes. The genes between these two points are exchanged to form two children.

4. Uniform Crossover Operator: This operator does not predetermine crossover points. Instead, a binary string of the same length as the chromosome is created. Wherever this binary string has a value of 1, the genes at that position are exchanged between the two parents, leading to the generation of two children.

5. Reduced Surrogate Crossover Operator: This operator restricts crossover operations to points where the genes of both parents are different. A random position is selected among these points to perform the crossover.

The MOBGA-AOS algorithm uses these five crossover operators, each with unique search abilities, to generate new solutions effectively. The adaptive operator selection mechanism in MOBGA-AOS maximizes the strengths of each operator to maintain both convergence and diversity in the search process. This approach demonstrates its effectiveness by producing feature subsets with fewer features and lower classification errors across various benchmark datasets.

All five crossover operators are utilized in the MOBGA-AOS algorithm, and the selection of each crossover operator is adaptive. This adaptive mechanism is a key feature of the algorithm. It allows the algorithm to dynamically choose which crossover operator to use at different stages of the evolutionary process, based on their performance in generating promising solutions.

The algorithm maintains a pool of crossover operators, and during the evolutionary process, it evaluates the effectiveness of each operator in producing offspring that are superior or at least comparable to their parents. This evaluation is done using reward and penalty vectors (nReward and nPenalty), which record the success or failure of each operator in each generation.

Based on this performance data, the algorithm adaptively selects the most suitable crossover operator for subsequent iterations. This approach ensures that the algorithm can effectively leverage the distinct search capabilities of each crossover operator, enhancing its ability to explore the solution space efficiently and to maintain a balance between exploration and exploitation.

This adaptive selection of crossover operators is a significant contribution of the MOBGA-AOS algorithm, as it enables the algorithm to adapt to the specific characteristics of the problem being solved, leading to more efficient and effective feature selection in classification problems.

The process of adaptive selection of crossover operators in the MOBGA-AOS algorithm involves a dynamic evaluation and selection mechanism based on the performance of each operator. This process can be summarized as follows:

1. Operator Pool and Performance Evaluation: The algorithm maintains a pool of crossover operators, each with unique search capabilities. During the evolutionary process, it evaluates the performance of these operators in generating offspring. This evaluation is based on how well the offspring compare to their parents in terms of fitness.

2. Reward and Penalty Mechanism: The algorithm uses two vectors, `nReward` and `nPenalty`, to record the success or failure of each operator in each generation. These vectors track the performance of the operators, rewarding those that produce promising offspring and penalizing those that do not.

3. Dynamic Selection Based on Performance: The algorithm adaptively selects which crossover operator to use in subsequent iterations based on the accumulated rewards and penalties. Operators that consistently generate better offspring (as indicated by higher rewards) are more likely to be chosen in future iterations.

4. Balancing Exploration and Exploitation: By using this adaptive mechanism, MOBGA-AOS effectively balances exploration and exploitation in the search process. Operators that are better at exploring the solution space might be chosen in the early stages of the algorithm, while those better at exploiting the current promising areas might be preferred in the later stages.

This adaptive operator selection mechanism is a key feature of the MOBGA-AOS algorithm, enhancing its ability to efficiently and effectively navigate the solution space for optimal feature selection in classification problems. The approach allows the algorithm to leverage the strengths of different crossover operators and to adapt to the specific characteristics and requirements of the problem at hand.

An "operator pool" in the context of genetic algorithms, like the MOBGA-AOS algorithm discussed in the paper, refers to a collection or set of different operators that are available for use during the evolutionary process. These operators are key components in genetic algorithms, responsible for creating variations and guiding the search towards optimal solutions. In MOBGA-AOS, the focus is on crossover operators, which are crucial for recombining genetic material from parent solutions to produce new offspring solutions.

Here's a more detailed explanation:

1. Variety of Operators: The operator pool contains different types of operators, each with unique characteristics and ways of manipulating the genetic material. In the case of MOBGA-AOS, the pool includes various crossover operators like single-point crossover, two-point crossover, uniform crossover, shuffle crossover, and reduced surrogate crossover. Each of these crossover operators has a distinct method of combining the genetic material from two parent solutions.

2. Purpose of the Operator Pool: The primary purpose of having an operator pool is to introduce diversity in the genetic search process. By having multiple operators, the algorithm can apply different strategies for creating offspring, which helps in exploring the solution space more effectively. This is particularly useful in complex optimization problems where the landscape of possible solutions is vast and varied.

3. Adaptive Selection: In adaptive operator selection, the algorithm dynamically chooses which operator to use from the pool based on their performance. This is done to enhance the efficiency of the search process. Operators that are more successful in producing fit offspring are chosen more frequently, while less successful ones are chosen less often. This adaptability allows the algorithm to respond to the changing landscape of the problem as the search progresses.

4. Role in Genetic Algorithms: In genetic algorithms, operators in the pool are used to manipulate the population of solutions (chromosomes). Crossover operators, for example, are responsible for combining parts of two parent chromosomes to create new offspring. This process is essential for introducing new genetic combinations into the population, which is critical for the evolutionary search for optimal solutions.

In summary, the operator pool in MOBGA-AOS is a set of different crossover operators that the algorithm can choose from, each contributing uniquely to the generation of new solutions. The adaptive selection mechanism from this pool is a key feature that enhances the algorithm's ability to efficiently explore and exploit the solution space in the context of feature selection in classification problems.

In the MOBGA-AOS algorithm, the evaluation of crossover operations based on how well the offspring compare to their parents in terms of fitness involves a systematic process. This process can be broadly outlined as follows:

1. Generation of Offspring: During each generation of the genetic algorithm, crossover operators from the operator pool are used to create offspring from parent solutions. Each crossover operator applies its unique method to combine genetic information from the parents.

2. Fitness Evaluation of Offspring: Once offspring are generated, their fitness is evaluated. The fitness typically reflects how well these offspring solutions meet the objectives of the problem. In the context of feature selection, this might include criteria such as classification accuracy and the number of features used.

3. Comparison with Parent Fitness: The fitness of the offspring is then compared to the fitness of their parents. This comparison is crucial to determine the effectiveness of the crossover operation that produced these offspring.

4. Reward and Penalty Mechanism:

- If the offspring are more fit or comparable in fitness to their parents, the crossover operator that generated these offspring is rewarded. This is done by incrementing a value in a reward vector (`nReward`) associated with that operator.

- Conversely, if the offspring are less fit than their parents, the operator is penalized by incrementing a value in a penalty vector (`nPenalty`).

5. Adaptive Selection of Operators:

- Based on the accumulated rewards and penalties, the algorithm adaptively adjusts the likelihood of each crossover operator being chosen in future generations.

- Operators that consistently produce fitter offspring (as indicated by higher rewards) become more likely to be selected, while those with more penalties are less likely to be chosen.

6. Iterative Process: This process of generating offspring, evaluating fitness, comparing with parents, and adjusting operator selection probabilities continues iteratively over multiple generations. The aim is to continuously refine the population of solutions towards the optimal set.

This methodology ensures that the crossover operations contributing most effectively to the evolutionary progress are favored, which in turn enhances the overall efficiency and efficacy of the genetic algorithm in finding optimal solutions. The adaptive nature of this process allows the algorithm to dynamically respond to the evolving solution landscape, maintaining a balance between exploring new areas of the solution space and exploiting known good areas.

The adaptive adjustment of the likelihood of each crossover operator being chosen in future generations in the MOBGA-AOS algorithm, based on the accumulated rewards and penalties, typically follows a process that involves updating the selection probabilities of the operators. Here's a general outline of how this might be implemented:

1. Tracking Operator Performance: Each crossover operator in the pool is associated with performance metrics, tracked through the reward and penalty vectors (`nReward` and `nPenalty`). These vectors record how often each operator produces offspring that are more fit (or at least as fit) as their parents (reward) and how often they produce less fit offspring (penalty).

2. Calculating Selection Probabilities:

- At the end of each generation, or at specified intervals, the algorithm calculates a score or weight for each operator based on its rewards and penalties.

- This can be done in various ways, but a common approach is to use a formula that increases the weight of an operator with more rewards and decreases it for those with more penalties. For example, the weight could be proportional to the ratio of rewards to the total of rewards and penalties.

3. Normalization of Weights: The calculated weights are then normalized so that they sum up to 1. This normalization converts the weights into probabilities.

4. Adaptive Selection:

- When the algorithm needs to select a crossover operator for generating new offspring, it uses these probabilities.

- Operators with higher probabilities (i.e., those that have been more successful in the past) are more likely to be selected, while those with lower probabilities are less likely.

- In the context of selecting crossover operators, the size of each slot on the wheel corresponds to the performance of the operator.

5. Feedback Loop: The success or failure of the operators in producing fit offspring continues to be tracked, and the rewards and penalties are updated accordingly. This creates a feedback loop where the selection probabilities are continuously adjusted based on the most recent performance of the operators.

6. Balancing Exploration and Exploitation: This adaptive mechanism ensures that the algorithm can dynamically balance exploration (trying out different operators) and exploitation (using operators that have been successful in the past). It allows the algorithm to effectively respond to the changing needs of the evolutionary process, making it more robust and efficient in finding optimal solutions.

By adaptively adjusting the selection probabilities of crossover operators based on their performance, the MOBGA-AOS algorithm can effectively leverage the strengths of various operators and adapt to the evolving solution landscape. This enhances its ability to explore the solution space efficiently and increases the likelihood of finding high-quality solutions to the problem at hand.

Sections 3.5.1, 3.5.2, and 3.6 of the paper "Adaptive Crossover Operator Based Multi-objective Binary Genetic Algorithm for Feature Selection in Classification" cover the following topics:

1. Section 3.5.1 - Credit Assignment Mechanism:

- This section describes the credit assignment mechanism used in the MOBGA-AOS algorithm. It explains how the algorithm assigns credits (rewards) or debits (penalties) to the different crossover operators based on their performance in generating offspring. The mechanism involves maintaining reward and penalty vectors (`nReward` and `nPenalty`) that track the success or failure of each operator in each generation.

2. Section 3.5.2 - Operator Selection Pool (OSP) Update:

- This section discusses how the Operator Selection Pool (OSP) is updated based on the rewards and penalties accumulated by the crossover operators. The OSP determines which operators are selected for generating new offspring in subsequent generations. The update mechanism is crucial for adapting the selection of operators to the evolving needs of the evolutionary process, enhancing the algorithm's efficiency.

3. Section 3.6 - Comparative Experiments:

- In this section, the authors present comparative experiments conducted to evaluate the performance of the MOBGA-AOS algorithm. They compare MOBGA-AOS with other well-known algorithms using various benchmark datasets. The section details the experimental setup, the datasets used, the evaluation metrics, and the results of these comparisons. The aim is to demonstrate the effectiveness and efficiency of the MOBGA-AOS algorithm in feature selection for classification problems.

These sections collectively detail the advanced mechanisms of the MOBGA-AOS algorithm, particularly focusing on how it dynamically adapts its operator selection to improve its search capabilities and the comparative analysis to validate its performance against other established algorithms.

The update of the Operator Selection Pool (OSP) in the MOBGA-AOS algorithm, based on the rewards and penalties accumulated by the crossover operators, involves a process where the performance of each operator influences its likelihood of being selected in future iterations. Here's a general overview of how this process might work:

1. Tracking Operator Performance: Each crossover operator in the OSP is associated with performance metrics, captured through reward and penalty vectors (`nReward` and `nPenalty`). These metrics reflect the operator's success in producing fit offspring (reward) and its failures (penalty).

2. Evaluating and Updating OSP:

- At regular intervals, the algorithm evaluates the performance of each operator based on the accumulated rewards and penalties.

- The evaluation might involve calculating a score or weight for each operator. This could be done, for example, by considering the ratio of rewards to the total of rewards and penalties, or through some other formula that quantifies performance.

3. Normalization and Probability Calculation:

- The scores or weights for each operator are then normalized to ensure they sum up to one. This normalization converts the scores into probabilities.

- These probabilities represent the likelihood of each operator being chosen from the OSP for the next generation of crossover operations.

4. Adaptive Selection of Operators:

- When generating new offspring, the algorithm selects crossover operators based on these updated probabilities.

- Operators that have higher probabilities (indicating better past performance) are more likely to be selected, while those with lower probabilities are less likely.

5. Feedback Loop for Continuous Update:

- The success of the operators in producing offspring in subsequent generations continues to be tracked with the reward and penalty system.

- This creates a feedback loop where the OSP is continuously updated, allowing the algorithm to adapt dynamically to the changing landscape of the optimization problem.

6. Balancing Exploration and Exploitation: The adaptive update of the OSP ensures a balance between exploring new crossover strategies (operators with lower probabilities might still be chosen occasionally) and exploiting known effective strategies (favoring operators with higher probabilities).

By regularly updating the OSP based on the performance of the crossover operators, the MOBGA-AOS algorithm ensures that it leverages the most effective operators for the problem at hand. This adaptive mechanism enhances the algorithm's ability to efficiently navigate the solution space and increases the likelihood of finding high-quality solutions.