A Frequent Pattern Mining Algorithm for Understanding Genetic Algorithms

Minh Nghia Le and Yew Soon Ong

Division of Information Systems, School of Computer Engineering
Nanyang Technological University, Nanyang Avenue, Singapore 639798

Abstract. In this paper, we present a Frequent Schemas Analysis (FSA) approach as an instance of Optinformatics for extracting knowledge on the search dynamics of Binary GA using the optimization data generated during the search. The proposed frequent pattern mining algorithm labeled here as *LoFIA* in FSA effectively mines for interesting implicit frequent schemas. Subsequently these schemas may be visualized to provide new insights into the workings of the search algorithm. A case study using the Royal Road problem is used to explain the search performance of Genetic Algorithm (GA) based on FSA in action.

1 Introduction

Details on how an evolutionary algorithm operates to search on a given landscape is unfortunately still not completely well understood. Knowledge about the search structure of the algorithm may bring new insights to *"how an algorithm search on a problem ?"* and/or *"under what circumstances does an algorithm functions well ?"*. Answers to these questions can be useful to assist one in selecting appropriate solvers for a given problem. In this paper, we particularly narrow our focus to Genetic Algorithm. Several attempts have been made in the past decades to enhance our understandings on the dynamics of GA and its connection to the problem landscape [1], [2]. To reduce the difficulty and complexity of the theoretical model in order to capture the stochastic behaviors of the optimization process, assumptions on infinite population size or use of simple operators are usually made in most studies. However, because of the assumptions made, most approaches cannot be used to analyze the behaviors of the general evolutionary search on a given problem at hand.

As introduced in [3], **optinformatics** refers to *the specialization of informatics for the processing of data generated in optimization so as to extract possibly implicit and potentially useful information and knowledge*. Even though it is generally tough to precisely predict how an evolutionary algorithm would search on a problem, the least one could do is to develop methods in optinformatics that aims to estimate the performance and highlight how the algorithm operates on the problem through the archived optimization data produced during (online) or at the end of the search (offline). In this paper, we present a Frequent Schemas Analysis (FSA) technique for extracting knowledge from the search process by using the historical optimization data, which are otherwise often discarded. FSA bringa about greater understanding of GA dynamics through mining for frequent schemas that exist implicitly within the optimization data. The rest of the

paper is organized as follows. Section 2 presents the definition of frequent schema and discusses the hardness of mining frequent schemas on GA data. In Section 3, we present the Frequent Schemas Analysis (FSA) technique for mining and visualizing interesting frequent schemas. Using the proposed FSA, we analyze the premature convergence experience of GA on the Royal Road problem in Section 5 and confirm the conclusion reported in earlier work. Finally, we conclude our findings in Section 6.

2 Frequent schemas in Genetic Algorithms

Over the past decades, many efforts on the analysis of Genetic Algorithm have been made since the first introduction of schemata and Schema Theorem by Holland. In his work, a schema is defined as a notational device to represent a set of fixed-length genomes (chromosomes) of length L having some alleles in common. It is a string of length L in which position i can take a specific allele value (i.e. 0 or 1 in case of binary string) or $*$ which represents a *don't care* symbol.

Let function $Freq(s, [m, n])$ denote the frequency of schema s in the populations over generations m to n. We define a schema s as **frequent schema** with a level θ in the period $[m, n]$ if and only if $Freq(s, [m, n]) \geq \theta$. One possible interpretation of a frequent schema s is that GA has spent at least θ percentage of its sampling budget on the hyperplane defined by s; or θ is a lower bound of the probability that a point in the hyperplane s is sampled by GA during the period $[m, n]$. Frequent schemas with a higher value of θ indicate regions with stronger convergence of GA in the period. However, a set of frequent schemas detected with higher value of θ is generally less specific (lower-order schemas) than those of lower values.

Note that each chromosome (binary string) in the data generated by GA in the period of generations can be transformed to a set of items (itemset). Each allele x_i at locus i $\in \Omega = \{0, 1\}$ is converted to an item with index $y_i = |\Omega| \times i + 1 + x_i$. Conversely, locus i and allele x_i can be calculated from y_i as $i = (y_i - 1)/|\Omega|$ and $x_i = (y_i - 1) mod |\Omega|$. For example, chromosome $010 \ldots 1$ of length L in an optimization is transformed to a set of items $\{1, 4, 5, \ldots, 2L + 2\}$. At this point, the term *frequent schema* and *frequent itemset* are used interchangeable, and the problem of finding frequent schemas on the aggregated GA data over a period can be considered as a problem in Frequent Pattern Mining [4].

Let $\mu_s(t)$ denote the observed average fitness of individuals that belong to schema s. As stated in Holland's book [2], *"..., if some schema begins to occupy a large fraction of the population (through consistent above-average performance), its rate of increase will come very close to* $[\mu_\xi(t)/\mu(t)] - 1$*"*, it is inferred that the frequencies of a schema with consistently *above-average* performance in a period form a *non-decreasing* sequence $\{Freq(s, t)\}_{t=m}^{n}$ and thus, a subset of consistently above-average schemas will become a part of the set of frequent schemas. With reference to [5], above-average schemas having short and closely spaced fixed positions that are not too unfavorably affected by crossover, termed as *building block*, tend to increase its frequency in the population as GA progresses, thus reducing the complexity of the problem. The Building Block Hypothesis (BBH) states that GA thus operates on these building blocks, growing them and mixing them with each other in an attempt to find the optimum solution. Since

low order building blocks will combine to form a higher order building blocks under BBH and building blocks contribute to the set of frequent schemas as discussed, it is expected that the length of most specific frequent schemas (itemsets) found will increase or *dense* datasets (i.e. data with long itemsets) will be obtained in the later period of the evolution. As discussed in [6], dense dataset often poses a challenge to the algorithms that mine for frequent patterns (or frequent schemas in this context). Thus, Building Block Hypothesis results in this hardness of mining frequent schemas from the dense data generated by GA in the later periods of the evolution.

3 Frequent Schemas Analysis

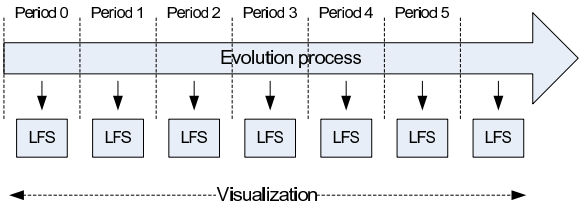

Fig. 1. Frequent Schemas Analysis

Instead of focusing on all details that may possibly be irrelevant in each search generation, the idea of Frequent Schema Analysis (FSA) is to capture relevant information about the schemata evolution for some periods of time through interesting frequent schemas as shown in Figure 1 in order to provide an overview or a sketch of the evolution process across periods.

Optimization data is first collected from the evolution process and divided into consecutive and non-overlapping periods. The sampling of GA in each period of the search space is analyzed by investigating on the set of frequent schemas ($Freq(s, P) \geq \theta$) found in that period. Alternatively, frequent schemas can also be compared across periods to understand the change in GA dynamics. Large value of θ gives more confidence on the located convergence regions but the frequent schemas are generally less specific, thus, interesting information may be not captured.

3.1 Frequent schemas mining

Besides requiring a huge amount of memory and computation resources, processing and storing all possible frequent schemas (e.g. at most 2^{30} for 30-bit strings) may not be necessary for the analysis and interpretation purposes. From the possibly numerous frequent schemas, it is up to the analyzer to select *interesting* schemas from the pool to investigate. In this paper, the *interestingness* metric is defined as the longest frequent schema/ itemset (LFS/ LFI in Figure 1) for that longest frequent schemas may provide a sketch on how GA progressively reduces the number of dimensions of its search space or biases its search towards promising convergence regions.

For this purpose, simply applying available algorithms in Frequent Pattern Mining may be very inefficient on some optimization data, such as those generated by the Royal Road problem which will be illustrated in Section 4. Based on the compact data structure *FP-tree* in *FP-growth* algorithm [6], we propose the *LoFIA* algorithm (Algorithm 1) which employs bottom-up and depth-first approach to quickly identify the *longest* frequent schemas from the optimization data. Support (frequency) counting for all frequent itemsets with prefix s is computed through the compact data structure FP-tree, instead of having to scan through the entire large dataset. FP-tree is reported in [6] to achieve good compactness in most of the time, especially in dense datasets such as GA optimization data.

Algorithm 1 Longest Frequent Itemset Mining Algorithm (LoFIA)

1: [$\mathbf{T}$] = *Build*(Data, θ) {Build FP-tree $\mathbf{T}$}
2: Set LongestFrequentItemsets (LFI) = **null**
3: Set $l_\theta = 0$
4: LFI = *LoFIA-Mining*(**null**, $\mathbf{T}$, LFI, l_θ)
5: **return** LFI

A search from a current tree T_s (i.e. FP-tree of prefix s) can be interpreted as finding the longest frequent itemset started with the prefix s. A conditional FP-tree $T_{s\alpha}$ is recursively generated from T_s which conforms to the depth-first search strategy (line 11, Algorithm 2). In this way, a complete set of frequent itemsets is found by recording a prefix as frequent itemset (line 9, Algorithm 2) and recursively generating the conditional FP-trees from the FP-tree of that prefix. The reader is referred to Han's paper [6] for the details of generating conditional FP-tree T_s and header table H_s.

Algorithm 2 LoFIA-Mining(s, $\mathbf{T_s}$, LFI, l_θ)

1: Build Header Table H_s
2: **for** each item α in H_s **do**
3: **if** $Order(\alpha|H_s) + |s| \geq l_\theta$ (*Criteria 2*) **then**
4: **if** $|s\alpha| > l_\theta$ **then**
5: *LFI.clear()*
6: $l_\theta = |s\alpha|$ {update to new expected longest length}
7: **end if**
8: **if** $|s\alpha| == l_\theta$ **then**
9: *LFI.insert*($s\alpha$)
10: **end if**
11: Build conditional FP-tree $T_{s\alpha}$ from T_s
12: **if** $|s\alpha| + D(T_{s\alpha}) \geq l_\theta$ (*Criteria 1*) **then**
13: *LoFIA-Mining*($s\alpha$, $T_{s\alpha}$, LFI, l_θ)
14: **end if**
15: **end if**
16: **end for**

The speed-up in the performance of *LoFIA* is achieved by incorporating the proposed pruning criteria of mining for longest frequent itemsets in order to avoid processing branches which are *already* known to contain only *short* frequent schemas. Furthermore, parameter l_θ is introduced in the algorithm as an expected lower bound of the maximum length of frequent itemsets at level θ (support threshold). It is worth noting that l_θ can be used to control a more effective pruning by initializing it to proper expected length of longest frequent itemset (> 0). When no prior knowledge about the length of the longest frequent itemset is available, the initial value of l_θ in the algorithm is 0. The pruning criteria used in the algorithm are based on two theorems which can be derived from the depth-first search strategy and the recursive method to construct conditional FP-tree T_s.

Theorem 1: *Denote $D(T_s)$ as the depth of the FP-tree T_s (i.e. length of the longest branch), the length of the longest frequent itemset with prefix s (i.e. L_s) is always less than or equal to $|s| + D(T_s)$:* $\mathbf{L_s} \leq |\mathbf{s}| + \mathbf{D(T_s)}$

Theorem 2: *Denote $Order(\alpha|H_s)$ as the order of item α in the conditional header table H_s, the longest frequent itemset with prefix $s\alpha$ has its length $L_{s\alpha}$* less than or equal to $Order(\alpha|H_s) + |s|$: $\mathbf{L_{s\alpha}} \leq \mathbf{Order}(\alpha|\mathbf{H_s}) + |\mathbf{s}|$

By these theorems, two pruning criteria in Algorithm 2 are derived as: (1) it is not necessary to proceed further on the branch of prefix s if $|\mathbf{s}| + \mathbf{D(T_s)} < \mathbf{l}_\theta$ (Theorem 1) and (2) when having the conditional FP-tree T_s of prefix s and an item α of interest, it is not necessary to proceed further on branch of prefix $s\alpha$ if $\mathbf{Order}(\alpha|\mathbf{H_s}) + |\mathbf{s}| < \mathbf{l}_\theta$ (Theorem 2). It is worth noting that different upper bounds for the length of the longest frequent itemset of certain prefix are used in Criteria 1 and 2 to reduce the computational cost in the mining process. In processing the branch with prefix $s\alpha$ which can be pruned by using either one of the two criteria, Criteria 2 requires less computational cost than Criteria 1 as its upper bound is based on the available information $Order(\alpha|H_s)$ instead of building the conditional *FP-tree* $T_{s\alpha}$. However, that Criteria 2 requires less computational cost comes as a trade-off to a tighter bound provided in Criteria 1 which is based on the depth of $T_{s\alpha}$.

3.2 Frequent schema visualization

A method used to visualize current convergence regions of GA in one period and the differences observed across periods serves to provide hints to the dynamics of GA. Vectors x of length L which represents the set of M most specific frequent schemas in consecutive periods are plotted against the time axis in the final visualization (Figure 3). The value of element x_i for locus i is then calculated by $x_i = \frac{N_1 - N_0}{M}$, where N_1 and N_0 are the number of schemas in the set has value 1 and 0, respectively, at locus i. Vector x can be considered as the combination of convergence regions in the probabilistic manner, in which $|x_i|$ near 1 indicates a clear overlapping of the interesting schemas at locus i ($x_i = 1/-1$ means all schemas agree on allele $1/0$, respectively). For example, vector x used to represent the list of the longest frequent schemas $\{111*00, 1*100*, *11000\}$ is $\{\frac{2}{3}, \frac{2}{3}, 1, \frac{-2}{3}, -1, \frac{-2}{3}\}$.

4 Computation cost and the effect of pruning in LoFIA

In this section, the runtime performance (i.e. total execution time) comparison among the *naive FP-growth* and *LoFIA* with different pruning criteria on test data was reported to confirm our comment on the hardness of GA optimization data and illustrate the effect of the pruning criteria in *LoFIA*. Note that the *naive FP-growth* algorithm mines for the longest frequent itemset by filtering the list of frequent itemsets found by *FP-growth* [6]. 10 sets of test data were generated from GA runs on 32-bit Royal Road problem as described in details in Section 5.

For each of the 10 data set, the experiment shows that the *naive FP-growth* could not complete processing the data within the limited runtime of 1 hour which is considerably larger than the runtime performance of our proposed algorithm (< 40 seconds).

The results show a significant speed-up of *LoFIA* over the *naive FP-growth* approach. Algorithm 2 using only Criteria 1 was also compared to *LoFIA* as shown in Figure 2. The figure suggests that as the computational cost of building conditional *FP-tree* is diminished by Criteria 2, the combined effect of pruning through both Criteria 1 and 2 in *LoFIA* helps to reduce the overall runtime significantly over the use of Criteria 1 alone.

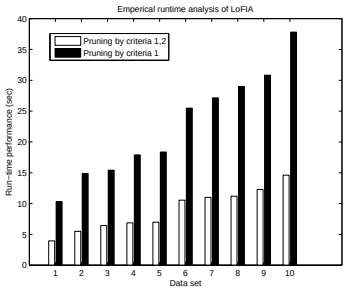

Fig. 2. Runtime analysis of pruning criteria in LoFIA

5 Frequent schema analysis of GA on Royal Road problem

Royal Road problem is one of the typical test functions for Binary GA, introduced in [7] by Mitchell *et al*. Two common configurations of linear (R1) or non linear reinforcement (R2) for the problem were described in [8]. Under the support of Building Block Hypothesis, the hierarchical structure of building blocks in Royal Road problem was expected to construct a *road* for GA to reach the global optimum. Even though the Royal Road problem was originally designed for GA to perform well, it is worth noting that the Random Mutation Hill Climbing (RMHC) described in [9] actually outperforms GA on the test function. Table 1 shows the average number of evaluations required by each algorithm to reach the optimal solution over 50 independent runs on the problem of $R2$ type, 32 bits ($K = 4$) and 64 bits ($K = 8$) in which K denotes the size of the fundamental blocks (e.g. s_1 to s_8 in Figure 3 for 32-bit problem, $K = 4$). GA is configured with one-point crossover p_{cross} of 0.8, bit-flip mutation p_{mut} of 0.003, fitness-proportional selection and $popsize$ of 50.

Methods	Royal Road (32K4)	Royal Road (64K8)
GA	7587.32 ± 7045.26	102880.96 ± 71723.45
RMHC	$\mathbf{412.22 \pm 206.61}$	$\mathbf{5876.86 \pm 2595.55}$

Table 1. Hill-climbing outperforms GA on Royal Road problem

To investigate what slowed down the GA on the Royal Road problem, Frequent Schemas Analysis (FSA) was used to analyze the archived optimization data of a GA search on the problem of 32 bits. Note that the optimal solution for this problem is $111\ldots1$ and the maximal fitness is 128. Here, the evolutionary search of 150 generations is divided into 10 periods, with each period consisting of 15 GA search generations. The most specific frequent schemas of each period at $\theta = 0.8$ were then obtained

using the proposed *LoFIA* algorithm. The convergence regions of GA (i.e. more than 80% of GA's sampling were on these regions/schemas) in each period are visualized in Figure 3.

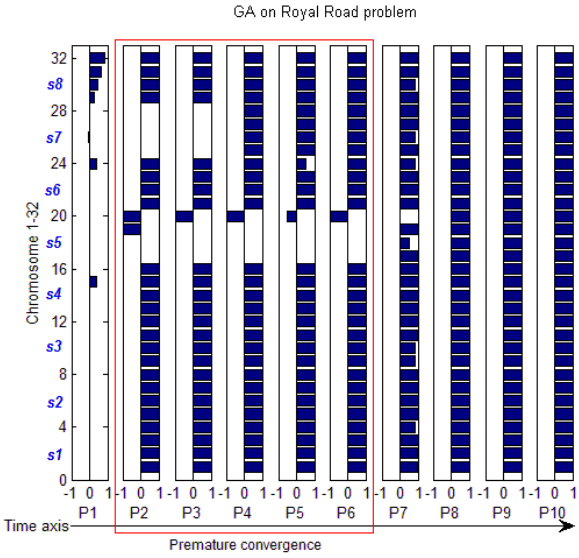

Fig. 3. Frequent schemas analysis of GA at $\theta = 0.8$

Firstly, the plot well illustrates the Building Block Hypothesis in which blocks are shown to be discovered and combined in reducing the complexity of the problem. As expected, the length of longest frequent schemas increases as evolution search progresses. In Figure 3, block s_5 of the frequent schemas was incorrectly identified in Period 2, containing two bits of 0 ($x_{20,19} = -1$). While other blocks of $1's$ of the convergence regions were correctly discovered by GA in early periods of the evolutionary process, the search had to spend a remarkable amount of time to identify the good configuration for block s_5 (approximately 5 periods P_2-P_6 or 75 generations) and then were able to converge to the global optimum. It is expected from the observation in Figure 3 that the block with incorrect alleles that hitchhikes in the previous generations requires significant efforts of the search before improvement in the schema is observed thus, suggesting a possible premature convergence of GA on the problem.

Here, the regions or schemas C defined by many correctly discovered blocks of $1's$ and one block with more than one bad alleles $0's$ (e.g. $C = [11\ldots1|*00*|1\ldots11]$) are considered as premature convergence regions of GA on the Royal Road landscape. An explanation might go as follows. We define the event of improvement of the search in this case as when an individual with *less* number of incorrect bits (i.e. bit 0) appears in the reproduction pool of the subsequent population. As one-point crossover now operates on the reproduction pool with a large number of individuals satisfying condition C, it becomes unlikely for the operator to bring about an event of improvement in this case. Furthermore, mutating other bits of 1 at the correct blocks of an individual will

decrease its fitness significantly due to the loss of building blocks. If the population falls in the premature convergence region with high probability, the mutated offspring with lower fitness will not be selected in the reproduction pool of the next generation. Therefore, it is expected that an event of improvement can only be achieved but at a small probability by mutating bit $0 \rightarrow 1$ while not changing other correct bits. Not to mention that the neutral selection pressure due to the plateau characteristic of the Royal Road fitness landscape defined on region C does not have any reinforcement on the event of improvement, i.e. an improved individual with less number of $0's$ can still be lost through selection. Hence, a series of improvement events which are required to identify the correct block (as more than one bad alleles $0's$ are presented) is strongly hindered by the combined effect of the above issues.

6 Conclusions

In this paper, a Frequent Schemas Analysis (FSA) technique is introduced as an instance of Optinformatics to provide a comprehensive picture of how the search process evolves, hence bringing new insights into the properties of GA search. In particular, the proposed *LoFIA* mining algorithm is employed in FSA to mine for interesting frequent schemas that exist implicitly within the archived Binary GA data. Through its pruning criteria, the proposed algorithm has shown a significant speed-up in performance over the *naive FP-growth* approach. The schemas of different search periods are then investigated via visualization method. Using the Royal Road problem, we demonstrated the ability of FSA in identifying the premature convergence of GA search which is validated in previous studies [8], [9].

References

1. De Jong, K.: Evolutionary Computation: A Unified Approach. MIT Press (2006)
2. Holland, J.: Adaptation in natural and artificial systems. MIT Press Cambridge, MA, USA (1992)
3. Le, M.N., Ong, Y.S., Nguyen, Q.H.: Optinformatics for schema analysis of binary genetic algorithms. Genetic and Evolutionary Computation Conference (GECCO) 2008
4. Pang-Ning Tan, Michael Steinbach, V.K.: Introduction to data mining. Pearson Addison Wesley (2006)
5. Goldberg, D.: Genetic algorithms in search, optimization, and machine learning. Addison-Westey, Readomg, Mass (1989)
6. Han, J., Pei, J., Yin, Y., Mao, R.: Mining frequent patterns without candidate generation: A frequent-pattern tree approach. Data Mining and Knowledge Discovery **8**(1) (2004) 53–87
7. Mitchell, M., Forrest, S., Holland, J.: The royal road for genetic algorithms: Fitness landscapes and ga performance. Toward a Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life **1001** (1992) 48109
8. Forrest, S., Mitchell, M.: Relative building-block fitness and the building-block hypothesis. In Whitley, L.D., ed.: Foundations of Genetic Algorithms 2. Morgan Kaufmann, San Mateo, CA (1993) 109–126
9. Mitchell, M., Holland, J., Forrest, S.: When will a genetic algorithm outperform hill climbing. Advances in Neural Information Processing Systems **6** (1994) 51–58