World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:3, No:12, 2009

Mining Sequential Patterns Using Hybrid Evolutionary Algorithm

Mourad Ykhlef, *Assistant Professor, King Saud University,* and Hebah ElGibreen, *Lecturer, King Saud University,*

Abstract—Mining Sequential Patterns in large databases has become an important data mining task with broad applications. It is an important task in data mining field, which describes potential sequenced relationships among items in a database. There are many different algorithms introduced for this task. Conventional algorithms can find the exact optimal Sequential Pattern rule but it takes a long time, particularly when they are applied on large databases. Nowadays, some evolutionary algorithms, such as Particle Swarm Optimization and Genetic Algorithm, were proposed and have been applied to solve this problem. This paper will introduce a new kind of hybrid evolutionary algorithm that combines Genetic Algorithm (GA) with Particle Swarm Optimization (PSO) to mine Sequential Pattern, in order to improve the speed of evolutionary algorithms convergence. This algorithm is referred to as SP-GAPSO.

Keywords—Genetic Algorithm, Hybrid Evolutionary Algorithm, Particle Swarm Optimization algorithm, Sequential Pattern mining.

I. Introduction

MINING Sequential Patterns in large databases has become an important data mining task with broad applications, including business analysis, web mining, security and bio-sequences analysis. It extracts patterns that appear more frequently than a user-specified minimum support while maintaining their item occurrence order. In this task, time is the most important factor, especially when the results are needed in a limited period of time.

Mining Sequential Pattern algorithms takes a long time to find the rules especially when they are applied on large databases. On the other hand, the evolutionary algorithms can find good Sequential Pattern rules within a short time. Nowadays, some evolutionary algorithms were proposed and have been applied in a timely manner. Genetic Algorithm (GA), which is an evolutionary algorithm, can be used to discover Sequential Pattern rules in a short time. It is a general purpose search algorithm which use principles inspired by natural genetic populations to evolve solutions to problems. In addition, Particle Swarm Optimization (PSO), which is also an evolutionary algorithm, can be considered as an alternative to the standard GAs. The PSO was inspired by insect swarms and has been shown to be a competitor to the GA for function optimization. The PSO has been applied widely in the function optimization, artificial neural networks' training, fuzzy control and some other fields. Since then several improved PSO algorithms have been developed.

Mourad Ykhlef is with King Saud University, College of Computer and Information Sciences, Information System Department, Kingdom of Saudi Arabia e-mail: (see ykhlef@ksu.edu.sa).

Hebah ElGibreen is with King Saud University, College of Computer and Information Sciences, Information Technology Department, Kingdom of Saudi Arabia e-mail: (see HJibreen@ksu.edu.sa).

Both GA and PSO algorithms have shown good performance for some particular applications but not for other ones. For example sometimes GAs outperformed PSO, but occasionally the opposite happened showing the typical application driven characteristic of any single technique. This is due to the different search method adopted by the two algorithms, the typical selection-crossover-mutation approach versus the velocity-position-global-local best communication.

Related Work: Existing approaches to find appropriate sequential sets for sequential pattern mining are mainly classified into two categories. The first is concerned with conventional methods and the other employs evolutionary based approaches. Under the first category, Agrawal and Srikant have introduced the Sequential Pattern mining problem [1], [2]. In addition, there were many fitness measures that were applied to be used in discovering Sequential Patterns, as in [26], [8], [15], and Piatesky-Shapiro [7] suggested some principles and property to choose the most appropriate measure depending on the problem. This paper developed an efficient approach and applied these principles to choose the appropriate measure for the proposed algorithm.

Differently, evolutionary based methods adjusted the time problem of conventional sequential methods according to an optimization scheme. Genetic Algorithm has been introduced in [7], [10], [9]. GA has many chromosome representations in [20], [23], [28], these representations have been studied and the most appropriate one has been chosen based on David advice in [9]. In [11], Kaya and Alhajj propose a novel multi-objective Genetic Algorithm method for optimizing quantitative fuzzy Sequential Patterns, their algorithm applied GA to discover fuzzy Sequential Patterns with a multi-objective measure. PSO, on the other hand, was introduces by Kennedy and Eberhart, 1995 [21], and its techniques have evolved greatly since then. This algorithm has been applied to association rules, clustering and classification, in [21], [14], [22], but not on Sequential Patterns. More recently, new approaches have been produced. These approaches combine more than one evolutionary algorithm to extract the rules. In [17], [16], they combined GA and PSO in order to cluster and classify rules. In our knowledge, no one yet tried to combine GA and PSO in the Sequential Pattern field.

In order to improve the speed of convergence of evolutionary algorithms, this paper will introduce a new kind of hybrid evolutionary method that consists of Genetic Algorithm (GA) and Particle Swarm Optimization (PSO). It will be referred to this algorithm by SP-GAPSO. This algorithm will be used to discover Sequential Pattern rules using the sequential interestingness measure [15] to reduce the time of algorithm taken to complete the process. The rest of this paper is organized as follows. Sequential Patterns, Genetic Algorithm, and Particle Swarm Optimization algorithms are defined in section 2, 3, and 4. The new approach of applying SP-GAPSO to Sequential Patterns is described in Section 5. Experimental results are reported in Section 6. At the end, the conclusion is presented in section 7.

II. SEQUENTIAL PATTERNS

Sequential Pattern mining addresses the problem of discovering the existent maximal frequent sequences in a given database. The problem was first introduced by Agrawal and Srikant [1], [2], where the basic concept involved in pattern detection has been established. It seeks similar patterns in data transaction; this approach is useful when the data to be mined has some sequential nature to deal with databases that have time-series characteristics, i.e. when each piece of data is an ordered set of elements [3]. For example, it can be said that 60% of patient who takes medicine X will take medicine Y afterward, regardless of the time gap.

Given a pharmacy database, where each transaction includes a patient ID, prescription time and its medicine, as in Table1, Sequential Pattern can be defined as follows.

Definition 1: Let $I = \{x1..xn\}$ be a set of items. An itemset is a non-empty subset of items, and an itemset with k items is called k-itemset. A sequence [1] $s = (X1..Xl)$ is an ordered list of itemsets, and an itemset $Xi(1 \leq i \leq l)$ in a sequence is called a transaction. In a set of sequences, a sequence s is maximal if s is not contained in other sequences. [11]

TABLE I
PHARMACY DATABASE

Medicine ID	Prescription Time	Patient ID
1	July 20, 2005	S9255
1	July 25, 2005	S7230
2	July 9, 2005	S3925, S8756
2	July 14, 2005	S9255
2	July 20, 2005	S3925, S8256, S7230
3	July 25, 2005	S8756, S8256
3	July 29, 2005	S9255

III. GENETIC ALGORITHM

Genetic Algorithm (GA) is general purpose search algorithm which use principles inspired by natural genetic populations to evolve solutions to problems [24].

All GAs typically starts from a set, called population, of random solutions (candidate). These solutions are evolved by the repeated selection and variations of more fit solutions, following the principle of survival of the fittest. The elements of the population are called individuals or chromosomes, which represent candidate solutions. Chromosomes are typically selected according to the quality of solutions they represent. To measure the quality of a solution, fitness function is assigned to each chromosome in the population. Hence, the better the fitness of a chromosome, the more possibility the chromosome has of being selected for reproduction and the more parts of its genetic material will be passed on to the next generations.

[1]Each sequence in sequential patterns is considered as a rule.

Genetic Algorithms are very easy to develop and validate, which makes them highly attractive, if they applied. The algorithm is parallel; it can be applied to large populations efficiently, so if it begins with a poor original solution it can rapidly progress to good solutions. Use of mutation makes the method capable of identifying global optimal, even in very difficult problem domains. The method does not require knowledge about the distribution of the data, this way GAs can efficiently explore the space of possible solutions. This space is called search space, and it contains all the possible solutions that can be encoded [6].

IV. PSO ALGORITHM

Particle Swarm Optimization (PSO) is swarm intelligence based algorithm to find a good solution to an optimization problem in a search space, or model and predict social behavior in the presence of objectives. It is a population-based evolutionary algorithm for problem solving. PSO algorithm was first represented by Kennedy and Eberhart, 1995 [21]. The techniques have evolved greatly since then, and the original version of the algorithm is barely recognizable in the current ones.

People solve problems by talking with other people about them, and as they interact their beliefs, attitudes, and behaviors change; the changes could typically be depicted as the individuals moving toward one another. The particle swarm simulates this kind of social optimization. The swarm is typically modeled by particles in multidimensional space that have a position and a velocity. These particles fly through hyperspace and have two essential reasoning capabilities: their memory of their own best position and knowledge of the global or their neighborhood's best.

As the swarm iterates, the fitness of the global best solution improves. It could happen that all particles being influenced by the global best eventually approach it, and afterward the fitness never improves despite iterated runs of PSO. The particles also move, in the search space, closely to the global best and not exploring the rest of search space. This phenomenon is called 'convergence'. If the inertial coefficient of the velocity is small, all particles could slow down until they approach zero velocity at the global best (illustrating bird swarm migration) [21].

V. MINING SEQUENTIAL PATTERNS USING SP-GAPSO

This section describes the SP-GAPSO algorithm for mining Sequential Patterns, associated with examples of pharmacy database (Table1). First, explanation of how SP-GAPSO algorithm represents the dataset and the chromosome structure, and how it encodes scheme and particles are shown. After that, description of genetic operators, and definition of fitness assignment and selection criteria are listed. Finally, the algorithmic structure along with its pseudo code is given.

A. Dataset Representation

Datasets contain data which the algorithm is applied to, in order to discover Sequential Patterns. To allow an efficient counting, a representation method must be implemented. In SP-GAPSO algorithm, vertical bitmap representation is used to represent the dataset. Vertical Bitmap Representation (VBR) can be defined as follow.

Definition 2 [4]: VBR efficiently stores the transactional database as a series of vertical bitmaps. A vertical bitmap is created for each item in the dataset, and each bitmap has a bit corresponding to each transaction in the dataset. If item i appears in transaction j, then the bit corresponding to transaction j of the bitmap for item i is set to one; otherwise, the bit is set to zero. To enable efficient counting and candidate generation, divide the bitmap such that all of the transactions of each sequence in the database will appear together in the bitmap.

For example, using the pharmacy database in Table1, the Vertical Bitmap Representation can be established as in Table2, where PID is patient ID and TID is the transaction ID.

TABLE II
DATASET REPRESENTATION

PID	TID	S3925	S7230	S8256	S8756	S9255
1	1	0	0	0	0	1
1	2	0	1	0	0	0
2	3	1	0	0	1	0
2	4	0	0	0	0	1
2	5	1	1	1	0	0
3	6	0	0	1	1	0
3	7	0	0	0	0	1

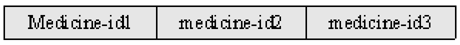
In VBR, if transaction T1 is before transaction T2 in a sequence, the index of the bit corresponding to T1 is smaller than the bit corresponding to T2. This propriety is important to know the order of the transactions without searching it.

B. Chromosome

This section discusses the used structure of GA chromosome, how it is represented in SP-GAPSO algorithm, and the associate particles of each chromosome.

1) Structure: SP-GAPSO algorithm uses a fixed length chromosome structure. For example, in pharmacy database, every prescription, given a medicine-id value, has been recorded to the database. These medicine-id values can be used for creating the chromosomes, and chromosome length can be set to the number of medicines prescribed to patients. If pharmacy database is searched, the chromosome's shape can be as Fig.1.

2) Representation: Before SP-GAPSO algorithm can be put to work on any problem, a method is needed to encode potential solutions to that problem in a form that a computer can process. In Genetic Algorithm there are many alternatives



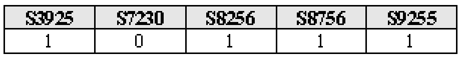
Medicine-id1	medicine-id2	medicine-id3

Fig. 1. Chromosome structure

to represent a chromosome based on other problem domains as described in [20], [23], [28]. To decide which representation is better to be used for Sequential Pattern rules David Goldberg [9] offered his advice saying *"The user should select a coding so that short, low-order schemata are relevant to the underlying problem and relatively unrelated to schemata over other fixed positions"* and *"The user should select the smallest alphabet that permits a natural expression of the problem"* [9]. The primary meaning behind these statements is that the proper choice of genetic representation is problem-dependent. If the application has a natural binary representation then binary is the best representation and if the application consists of integer variables then an integer representation may be appropriate.

In SP-GAPSO algorithm, binary is the most suitable representation because, comparing to the integer representation which lost its advantage of canceling the encoding phase, it represent the needed information in less space (element occurred or not).

For example, using the pharmacy database in Table1, a sequence $<(S3925, S8256)(S9255)(S9255, S3925, S8756)>$ can be represented as in Fig.2.



S3925	S7230	S8256	S8756	S9255
1	0	1	1	1

Fig. 2. Chromosome representation

Additionally, because of the unusual structure of Sequential Pattern rules, chromosomes representation became an issue. It is important in Sequential Pattern to encode the sequence of transactions. But, as you can see in Fig.2, you cannot extract the sequence order directly. To solve this problem it has been decided to associate the transactions sequences as metadata with each chromosome. This solution will take less time and space to complete the algorithm as a whole.

For example, using the pharmacy database in Table1, the metadata of sequence $<(S3925, S8256)(S9255)(S9255, S3925, S8756)>$ that is represented in Fig.2 can be indexed as in Fig.3.

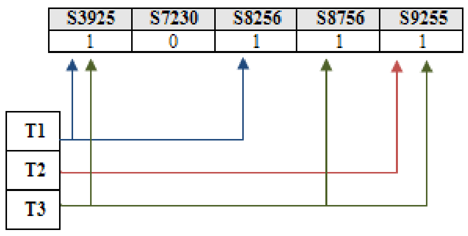


Fig. 3. Metadata representation

3) Particles: Particles in SP-GAPSO have position and velocity. Additionally, these particles have two essential reasoning capabilities: their memory of their own best position and knowledge of the global best position. In SP-GAPSO algorithm, every chromosome is associated with a particle. Each Particle variables can be defined as follows.

Definition 3 [12], [27], [5]: Given that the problem of Sequential Patterns is to maximize the fitness, each particle in SP-GAPSO have four variables (v, x, p_i, p_g). It will update its position and velocity using (1) and (2), and its local and global best position using (3) and (4), as follow.

$$v_{i+1} = [W * v_i + \phi 1 \otimes (p_i - x_i) + \phi 2 \otimes (p_g - x_i)] Mod 1 \quad (1)$$

$$x_{i+1} = [x_i + v_{i+1}] Mod 1 \quad (2)$$

$$p_{i+1} = \begin{cases} p_i & \text{if } p_i \geq x_{i+1}; \\ x_{i+1} & \text{if } p_i < x_{i+1}; \end{cases} \quad (3)$$

$$p_g = \begin{cases} p_g & \text{if } p_g \geq p_i; \\ p_i & \text{if } p_g < p_i; \end{cases} \quad (4)$$

Where i = [1 ... MaxPopulation], and W is called inertia weight, which used to eliminate the necessity of checking if v is inside its range. Eberhart and Shi [5] suggested the use of a time-varying inertia weight, gradually decreasing (in this case increasing) its value typically from 0.9 to 0.4. Variable v_i denote the velocity of the ith particle in the swarm, x_i denote its position, p_i denote the personal best position and p_g denote the best position found by particles in its neighborhood. $\phi 1$ = (c1R1) and $\phi 2$ = (c2R2), where R1 and R2 are random numbers, uniformly distributed in [0 .. 1], c1 and c2 are positive constants called acceleration coefficients where their summation should equal to four. The symbol ϕ denotes pointwise vector multiplication.

SP-GAPSO algorithm uses the Modulus operation[2] with (1) and (2). It will use Mod operation with 1 in order to insure that it will not exceed the range of the fitness function, which is [0 .. 1].

C. Genetic Operators

Genetic Algorithm uses genetic operators to generate the offspring of the existing population. This section describes three operators of Genetic Algorithms that were used in SP-GAPSO algorithm: selection, crossover and mutation.

1) Selection: The selection operator chooses a chromosome in the current population according to the fitness function and copies it without changes into the new population. SP-GAPSO algorithm used Elitist selection, where the fittest members of each generation are copied into the next generation.

2) Crossover: The crossover operator, according to a certain probability, produces two new chromosomes from two selected chromosomes by swapping segments of genes. SP-GAPSO algorithm used single-point crossover operation with probability 0.7; chromosomes can be created as in Fig.4 [19].

[2]For example (1.5 Mod 1 = 0.5).

3) Mutation: The mutation operator is used for maintaining diversity. During the mutation phase and according to mutation probability, 0.001 in SP-GAPSO algorithm, value of each gene in each selected chromosome is changed, as in Fig.4 [18].

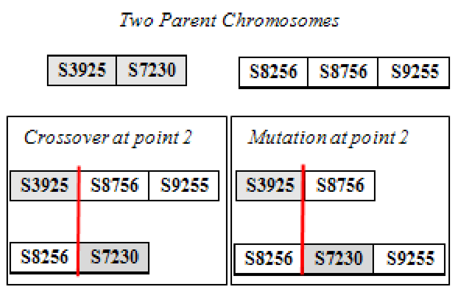


Fig. 4. Schematic representation of crossover and mutation operators

D. Fitness Function

The relationships in Sequential Patterns are resulting from applying some measures to determine and generate rules, called fitness function. There are a lot of evaluation rule measures in [26], [8], [15] that comes from statistics, machine learning and data mining, each of them trying to evaluate one feature of the rule (precision, interest, reliability, comprehension, simplicity, etc.). Interestingness measures play an important role in data mining regardless of the kind of patterns being mined.

Piatesky-Shapiro, 1991 [20], proposed three principles that obeyed by any objective measure. Lenca et al. [20], proposed five properties, based on Piatesky principles, to evaluate the measures. In SP-GAPSO algorithm, after using these principals and property, "Sequential interestingness" measure [15] is the most appropriate measure to be used since it applies most of the requirement; it can be defined as follows.

Definition 4 [15]: The sequential interestingness of a rule s can be generated from (5), where $(\alpha \geq 0)$ is a parameter defined by the user that represents how important the frequency of the pattern is and s_p is the transactions of s.

$$inst(s) = min_{s_p \in s}\{(Conf(s_p|s))^{\alpha}\} * Supp(s) \quad (5)$$

The first term of the equation evaluates that the frequencies of the sub-patterns, i.e. transactions, are not frequent while the second term evaluates that the frequency of the pattern is frequent[3] . Parameter α is called the confidence priority, and the pattern that is bigger than or equal to the minimum sequential interestingness given by the user is called the interesting pattern.

E. SP-GAPSO Algorithm for Mining Sequential Patterns

This section presents the algorithmic structure of SP-GAPSO. Then pseudo code of the algorithm will be described.

[3]As well known, $Conf(s_p|s)$ divide number of s and s_p occurrence, in the database, over number of s occurrence; while Supp(s) divide number of s occurrence, in the database, over number of all sequences available in the database [1, 2].

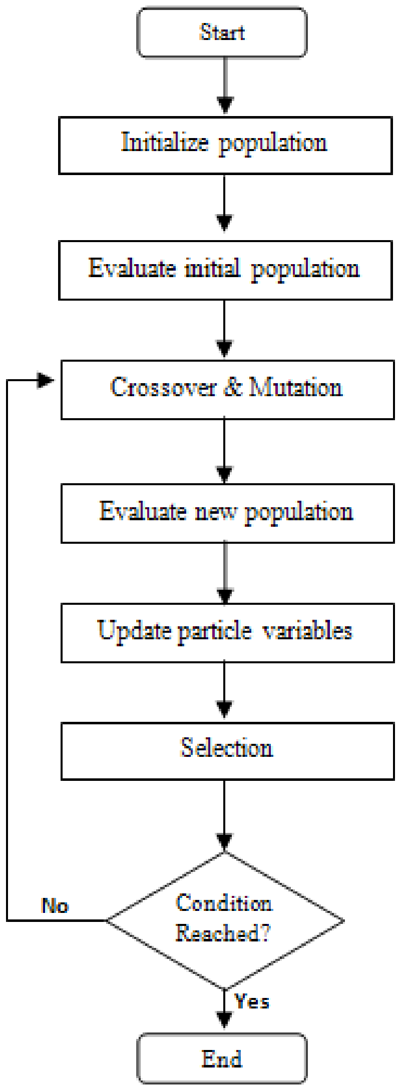
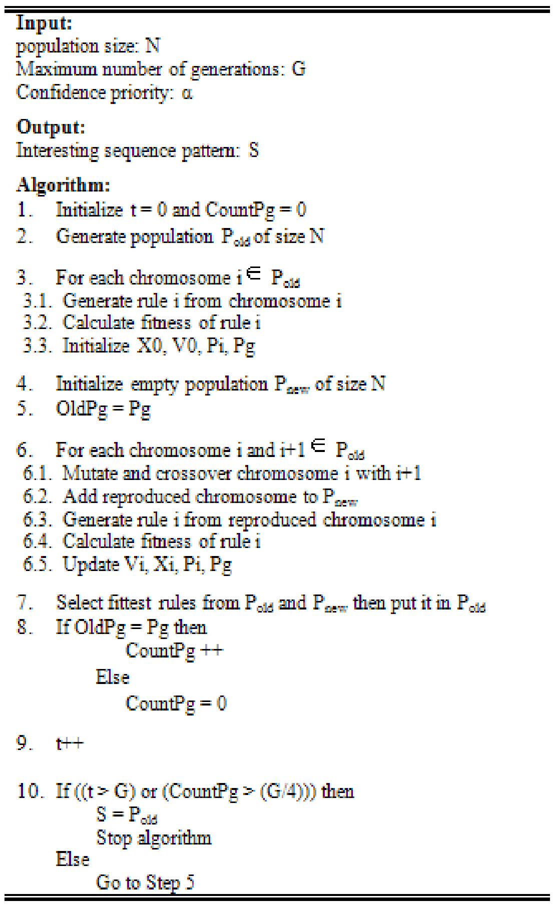


Fig. 5. SP-GAPSO flowchart



```
Input:
population size: N
Maximum number of generations: G
Confidence priority: α

Output:
Interesting sequence pattern: S

Algorithm:
1.   Initialize t = 0 and CountPg = 0
2.   Generate population P_old of size N

3.   For each chromosome i ∈ P_old
 3.1. Generate rule i from chromosome i
 3.2. Calculate fitness of rule i
 3.3. Initialize X0, V0, Pi, Pg

4.   Initialize empty population P_new of size N
5.   OldPg = Pg

6.   For each chromosome i and i+1 ∈ P_old
 6.1. Mutate and crossover chromosome i with i+1
 6.2. Add reproduced chromosome to P_new
 6.3. Generate rule i from reproduced chromosome i
 6.4. Calculate fitness of rule i
 6.5. Update Vi, Xi, Pi, Pg

7.   Select fittest rules from P_old and P_new then put it in P_old
8.   If OldPg = Pg then
             CountPg ++
         Else
             CountPg = 0

9.   t++

10.  If ((t > G) or (CountPg > (G/4))) then
         S = P_old
         Stop algorithm
     Else
         Go to Step 5
```

Fig. 6. SP-GAPSO pseudo code

1) Algorithmic Structure: The flow chart of SP-GAPSO algorithm is shown in Fig.5. After the encoding of the dataset, using the vertical bitmap representation, the algorithm starts by initializing the population. Then its fitness value is determined for each chromosome using (5). Subsequently, the following processes are repeated until the pre-specified maximum number of generations is achieved or the global best haven't improved for a known number of times.

First, existing chromosomes are used to generate new ones by applying crossover and mutation operators. Then, the fitness value is determined for each new chromosome in addition to their particle variables $(v, x, p_i, and\ p_g)$. At the end, chromosomes survive based on their fitness that is used in the process. This way, the interesting set is determined and the target is achieved.

2) Pseudo code: The pseudo code of SP-GAPSO is shown in Fig.6. First, population size, maximum number of generations, and confidence priority will be taken as input. Then SP-GAPSO algorithm will work as follow.

1) Loop counter and counter of global best improvement is set to zero.
2) Initial population is generated.
3) Every valid chromosome (with active elements, i.e. not only zeros) is associated with a sequential pattern rule, as described in section V.B.2. Then, it calculate the rule fitness (using (5)) and particle variables (velocity and position are set to zero while best local and global are set using (3) and (4), respectively).
4) Second population P_{new} is initialized with zeros; it will contain P_{old}'s chromosomes after applying GA operators, i.e. its children.
5) Preserve global best, to be compared to the new one, in order to decide whether it improved or not.
6) Population P_{old} is mutated and crossed over and then the resulting children is put into P_{new}, to associate the appropriate sequential pattern rule, as described in section V.B.2, and calculate its fitness and particle variables value using (5), (1), (2), (3), and (4), respectively. Remember that (1) and (2) use Modulus operator to preserve the fitness range, as described before in Definition 3. This step preserves the children of the population P_{old} in order to be compared in the selection step.
7) Selection operation is conducted, between P_{old} and

P_{new}, and the best rules chosen between the parents in P_{old} and the children in P_{new} are placed into P_{old}.

8) Update CountPg according to the value of P_g. If the global best did not improve after the selection between parents and children, i.e. between P_{old} and P_{new}, then the counter will be increased. This step takes advantage of PSO in order to reduce the time.
9) Increase the loop counter.
10) Check the termination condition. If the maximum generation number is reached or when the global best is not improving for a specific number of times, i.e. CountPg variable equals quarter maximum generation number, then stop the algorithm and put the P_{old} into S, which have the fittest rules conducted from the algorithm; otherwise, go to step 5.

As you can see from this algorithm, it took advantages of GA operators, i.e. mutation, crossover and selection, in addition to PSO variables, i.e. velocity, position, local and global best, to increase the knowledge of solution space and decrease the time taken to complete the process.

VI. Experiment Results

This section reports the results of the experiments conducted to analyze a pharmacy database. These results, as in the graphs below, are not based on a single run of each algorithm; instead it is averaged over many runs.

Pharmacy database has been used because medication management is an important process especially in pharmacy field. In this process, prescribing errors might occur and place patients at risk of adverse drug events. Prescribing errors are a particular concern for which conflicting with another prescribed medicine could cause severe harm for a patient. The knowledge of prescriptions can reduce the risk of harm to patients from prescribing errors. Extracting these knowledge will give the pharmacist general awareness of prescriptions that will alert him when an unusual activity occur in order to check again with the doctor. Additionally, due to the shortage of pharmacists there is urgency for efficiency improvement in pharmacy operations, this improvement is also important to contain the cost of healthcare delivery. SP-GAPSO can be applied in order to be used in discovering the needed knowledge in a timely manner.

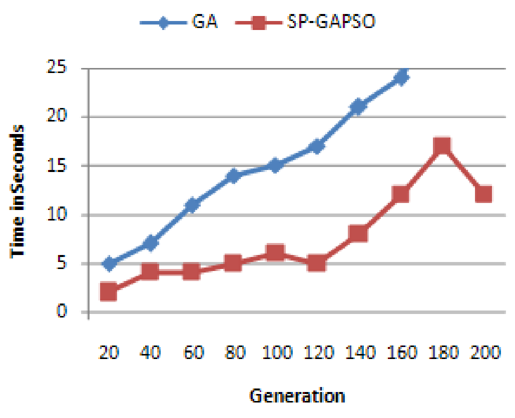
All the experiments were performed on 3 GHz Intel Pentium 4 PC machine with 1.50 GB RAM, running Microsoft Windows XP. The algorithm was written with java in Borland JBuilder environment. During all of the tests, confidence priority, population, and generation were given by the user.

As experimental data, a real pharmacy database is taken from King Faisal Specialist Hospital and Research Centre, in KSA. The experiment was done on one year of heart patients' prescriptions with 1361 transaction and 50 patients. The crossover and mutation probabilities were 0.7 and 0.001, respectively. The output of this experiment is a file that includes the interesting rules that represent the most suitable prescriptions sequences, along with its fitness and the time of the process. For example, taken from the output file, a sample of the rules is as follows.

R1: $[S7230] \rightarrow [S6825, S8756](F = 0.66)$
R2: $[S7230][S7230] \rightarrow [S8756](F = 0.70)$

R1 warn that when 100MG ASPIRIN is prescribed for patients, (67%) of these patients will take 5MG RENITEC with 50MG NORMOTEN afterwards, while R2 warn that when 100MG ASPIRIN are prescribed two times in a row for patients, (70%) of these patients will take only 50MG NORMOTEN afterwards.

In the experiment, there are three parameters that must be determined: number of generation, population size, and confidence priority. Two experiments have been conducted in order to compare the speed and fitness of SP-GAPSO with GA only, and show the improvement that SP-GAPSO has brought. First experiment set the population to 20 and the confidence priority to 0.5 with generation of [20...200]. The second, set the generation to 100 and the confidence priority to 0.5 with population of [20...200]. Both experiments were done on SP-GAPSO and GA.



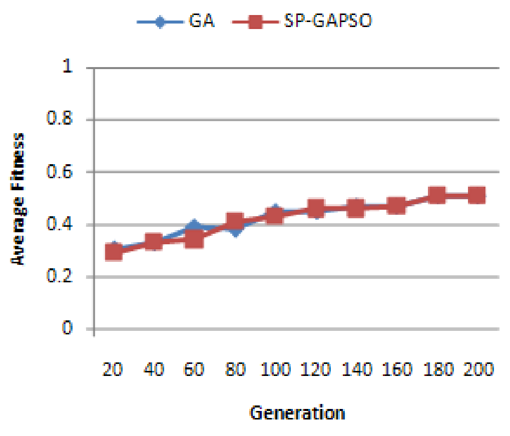
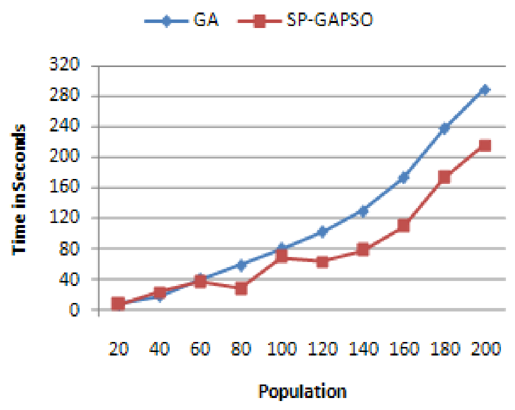


Fig. 7. Time and fitness related to generation

Fig. 7 shows the best result of the first experiment, in both SP-GAPSO and GA, related to increasing the generations. It shows the time, in seconds, spent by both algorithms and the average fitness of their final output. Comparing the result of GA and SP-GAPSO you can see that SP-GAPSO take less time than GA, while they almost have the same average fitness.

In addition, increasing the generation will increase the average fitness; however this might also increase the execution time, especially in GA since it only depends on the generation to end the execution.

Fig. 8 shows the best result of the second experiment, also in both SP-GAPSO and GA, related to increasing the population size. It shows the time, in seconds, spent by both algorithms and the average fitness of their final output. Comparing the result of GA and SP-GAPSO you can see that it has similar effect as the first experiment, SP-GAPSO take less time than GA while they almost have the same average fitness; and increasing the population will increase the execution time, but it might aggravate the fitness.



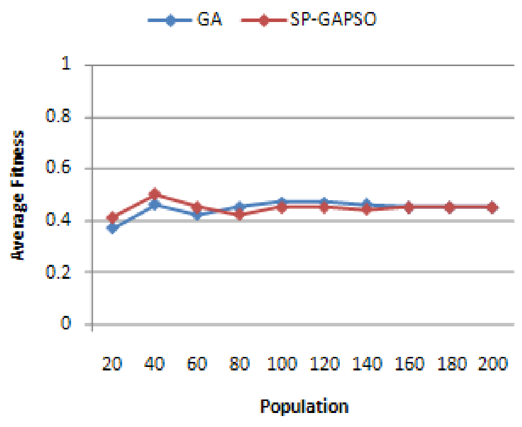


Fig. 8. Time and fitness related to population size

In addition, comparing the two experiments in Fig. 7 and 8, you can see that increasing population size will slow down GA and GAPSO more than in increasing the generations. As for the fitness, increasing the generation will improve it but increasing the population, at first, will improve it but then it will not change or even get it worse.

From the experiment, it is observed that SP-GAPSO has better performance than GA, especially in the time spent to extract the fittest rules. Moreover, increasing of generation will take less time than increasing of population size. But, either ways, SP-GAPSO doesn't take along time; it is only a matter of seconds. In addition, increasing the population will not guarantee improvement of average fitness, some times it even make it worse, while increasing the generation will most likely give better fitness on average.

At the end, from the results given above, it can be seen that it is better to use SP-GAPSO than only GA. Moreover, even if SP-GAPSO reduces the time, it is important to choose the right generation and population size, which means that you have to be careful in choosing the generation and population size. In our opinion, it is better to use SP-GAPSO with moderate population size and high generation to extract sequential patterns. This tradeoff decision will prospectively guarantee good rules in short time.

VII. Conclusion

In this paper, hybrid evolutionary algorithm has been applied, combining genetic and Particle Swarm Optimization algorithms, to find frequent sequences in sequential dataset. The algorithm utilizes the property of evolutionary algorithms that discovers new rules in a short time to overcome the shortage of conventional sequential algorithms. The use of mutation in Genetic Algorithm makes the method capable of identifying global best, even in very difficult problem domains. The method does not require knowledge about the distribution of the data, this way SP-GAPSO can efficiently explore the space of possible solutions. SP-GAPSO algorithmic components including binary representation, selection, crossover/ mutation operators, and particles variables, all contribute to this excellent run time. Experimental results demonstrated that SP-GAPSO algorithm, in Sequential Patterns, is more appropriate than GA, since they are similar in fitness while SP-GAPSO improved the time. In addition, you have to be careful in choosing the right generation and population size in order to get the best performance.

Acknowledgment

The authors acknowledge the cooperation of King Faisal Specialist Hospital and Research Centre, in KSA, for providing a real pharmacy database for this work experiment.

References

[1] Agrawal R. and Srikant R. *Mining Sequential Patterns*. IBM Almaden Research Center, 650 Harry Road, San Jose, CA 95120-6099.
[2] Agrawal R. and Srikant R. 1996. *Mining Sequential Patterns: Generalizations and Performance Improvements*. IBM Almaden Research Center, 650 Harry Road, San Jose, CA 95120.
[3] Antunes C. and Oliveira A. *Sequential Pattern Mining Algorithms: Trade-offs between Speed and Memory*. Instituto Superior Tcnico / INESC-ID.
[4] Ayres J. Gehrke J. Yiu T. and Flannick J. 2002. *Sequential Pattern Mining using a Bitmap Representation*. SIGKDD '02 Edmonton, Alberta, Canada.
[5] Blum C. and Li X. 2008. *Swarm Intelligence in Optimization*. Natural Computing Series. Springer-Verlag Berlin Heidelberg, 43-85.
[6] Colombetti M. and Dorigo M. 1993. *Training Agents to Perform Sequential Behavior*. Italian National Research Council, TR-93-023.
[7] Fayyad U. Piatetsky-Shapiro G. and Smyth P. 1996. *From Data Mining to Knowledge Discovery in Databases*. American Association for Artificial Intelligence: AI Magazine, 37-54.
[8] Geng L. and Hamilton H. 2005. *Interestingness Measures for Data Mining: A Survey*. Computer Science, University of Regina, Regina, Saskatchewan, Canada.

[9] Goldberg D. 1989. *Genetic Algorithms*. Addison Wesley, ISBN: 0-201-15767-5.

[10] Herrera F. Lozano M. and Verdegay J. 1998. *Tackling Real-Coded Genetic Algorithms: Operators and tools for the Behaviour Analysis*. Artificial Intelligence Review, Vol.12, 256-319.

[11] Kaya M. Alhajj R. 2004. *Multi-Objective Genetic Algorithm Based Approach for Optimizing Fuzzy Sequential Patterns*. 16th IEEE International Conference on Tools with Artificial Intelligence, 1082-3409/04.

[12] Montes M. 2007. *Particle Swarm Optimization*. IRIDIA-CoDE, Universite Libre de Bruxelles (U.L.B.).

[13] Pakhira M. and De R. 2007. *Generational PipeLined Genetic Algorithm (PLGA) using Stochastic Selection*. International Journal of Computer Systems Science and Engineering, Vol.4, No.1, 75 - 88.

[14] Premalatha K. and Natarajan A. 2009. *Procreant PSO for fastening the convergence to optimal solution in the application of document clustering*. CURRENT SCIENCE, Vol. 96, No. 1, 137- 143.

[15] Sakurai S. Kitahara Y. and Orihara R. 2008. *A Sequential Pattern Mining Method based on Sequential Interestingness*. International Journal of Computational Intelligence, 252-260.

[16] Shi X, Lu Y. Zhou C. Lee H. Lid W. and Liang Y. 2003. *Hybrid Evolutionary Algorithms Based on PSO and GA*. IEEE, 2393- 2399.

[17] Shi X, Wan L. Lee H. Yang X. Wang L. and Liang Y. (2003). *An Improved Genetic Algorithm With Variable Population Size and a PSO-GA Based Hybrid Evolutionary Algorithm*. Proceedings of the Second International Conference on Machine Learning and Cybernetics, 1735-1740.

[18] Spears W. *Crossover or Mutation?*. Navy Center for Applied Research in Artificial Intelligence, Naval Research Laboratory, Washington, D.C. 20375-5320.

[19] Spears W. and Anand V. *A Study of Crossover Operators in Genetic Programming*. Navy Center for Applied Research in AI, Washington, D.C 20375-5000.

[20] Tay J. and Wibowo D. 2004. *An Effective Chromosome Representation for Evolving Flexible Job Shop Schedules*. Intelligent Systems Lab, Nanyang Technological University, LNCS 3103, 210-221.

[21] Toracio A. Pozo A. 2007. *Multiple Objective Particle Swarm for Classification-Rule Discovery*. IEEE Congress on Evolutionary Computation (CEC), 684- 691.

[22] Wang H. Yeh W. Huang P. and Chang W. 2009. *Using association rules and particle swarm optimization approach for part change*. Expert Systems with Applications, Vol. 36, 8178-8184.

[23] Wannarumon S. *Aesthetic Creation of Endless Forms: An Application in Jewelry Design*. Naresuan University, Thailand, 395- 410.

[24] Wook J. and Woo S. 2005. *New Encoding/Converting Methods of Binary GA/Real-Coded GA*. IEICE Trans, Vol.E88-A, No.6, 1545-1564.

[25] Ykhlef M. and El-Gibreen H. 2009. *Mining Sequential Patterns in Pharmacy Database Using Genetic Algorithm*. 4th International Conference on Broadband Communication, Information Technology and Biomedical Applications BroadBandCom '09, Wroclaw, Poland.

[26] Zhao Q. and Bhowmick S. 2003. *Sequential Pattern Mining: A Survey*. CAIS, Nanyang Technological University, Singapore, No. 2003118, 1-27.

[27] Zhanga J. Huanga D. Lokd T. Lyue M. 2006. *A novel adaptive sequential niche technique for multimodal function optimization*. Neurocomputing, Vol. 69, 2396-2401.

[28] Zhou Y. 2006. *Study on Genetic Algorithm Improvement and Application*. Master thesis.