An Adaption of the Schema Theorem to Various Crossover and Mutation Operators for a Music Segmentation Problem

Brigitte Rafael
brigitte.rafael@heuristiclab.com

Michael Affenzeller
michael.affenzeller@fh-hagenberg.at

Stefan Wagner
stefan.wagner@fh-hagenberg.at

University of Applied Sciences Upper Austria
School of Informatics, Communications and Media
Heuristic and Evolutionary Algorithms Laboratory
Hagenberg, Austria

ABSTRACT

The schema theorem provides theoretical background for the effectiveness of genetic algorithms and serves as a formal model to explain their success. It describes the functionality of genetic algorithms under very restrictive limitations of a canonical genetic algorithm which applies a binary alphabet, individuals of equal length, fitness-proportional selection, single-point crossover, and gene-wise mutation. Applications of genetic algorithms, however, are often based on noncanonical variations and, therefore, are not verified by the theory of the traditional theorem. This paper describes the adaption of the theorem for various other crossover and mutation operators focusing on the application of genetic algorithms to a music segmentation problem.

Categories and Subject Descriptors

I.2.8 [**ARTIFICIAL INTELLIGENCE**]: Problem Solving, Control Methods, and Search—*Heuristic methods*

Keywords

genetic algorithms, schema theorem, building block hypothesis, Music Information Retrieval

1. INTRODUCTION

Segments of a music segmentation can start at any arbitrary position of the composition. The high complexity results in an exponential increase of runtime for longer compositions. Therefore, it is not possible to evaluate all potential segmentations but a solution of sufficient quality has to be found in reasonable time. Given those circumstances, the problem domain of music segmentation seems to be highly suited for applying genetic algorithms (see [3, 12]). This paper provides the theoretical background for the application of genetic algorithms to the music segmentation problem based on the schema theorem.

The schema theorem was introduced by Holland [4] and, later on, Goldberg [2] to provide a theoretical explanation for the good performance of genetic algorithms. The theorem is widely acknowleged but has also been criticized for two reasons: It only holds for the canonical genetic algorithm and it focuses on a worst case scenario. Plenty of work has been done by various researchers like Stephens and Waelbrock [15, 16, 17], Whitley [18], Wright [20], Altenberg [1], Poli [8, 9, 10], and others to overcome those limitations. This paper focuses on the adaption of the schema theorem for the music segmentation problem based on former research of Whitley [18].

In the section following the introduction the reader is introduced to the music segmentation problem. The third section describes the application of genetic algorithms to this problem. Section four focuses on the schema theorem and its adaption to various crossover and mutation operators. It also provides an example for the development of schemata in a genetic algorithm applied to the music segmentation problem. The last section summarizes and concludes the paper.

2. THE MUSIC SEGMENTATION PROBLEM

During the last years plenty of research has been done in the field of music information retrieval (MIR), also including various aspects of music segmentation. Providing structural information for a composition is crucial for several tasks of MIR. Music segmentation targets at the identification of boundaries between structurally relevant parts. There are two main aspects of music segmentation: boundary detection and pattern detection. Orio and Neve [7] give an introduction to both approaches as well as the results of experiments using either of them. An approach to combine both methods is described in [5].

Whereas boundary detection algorithms need extraopus information and are limited to music complying with existing domain knowledge, pattern discovery algorithms can be applied to a broader musical spectrum. As a downside, pattern discovery algorithms are not successful in segmenting music that does not contain any repeating patterns. The

GECCO'12 Companion, July 7–11, 2012, Philadelphia, PA, USA.

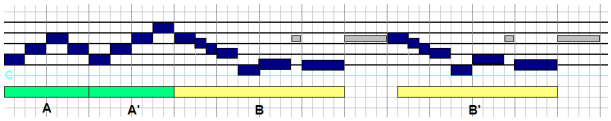

Figure 1: Segmentation

approach presented in this paper applies pattern detection as it concentrates on music with a high repetition factor.

Music data is represented in the MIDI (Musical Instrument Digital Interface) format. For each note MIDI files must save some information about the instrument that it is meant for. Therefore, data in MIDI files is divided into different tracks. Each track can be used for one instrument or several tracks may express various voices of the same instrument. This concept provides an advantage over audio formats in analyzing music. Each track can be analyzed separately and track-related patterns can be found independently of the other tracks.

The segmentation of a track contains a list of segment groups that indicate where parts of the track are repeated. It therefore gives an insight into the internal structure of a track. All segments of the segment groups of a segmentation must be distinct, which means that no overlap between any pair of consecutive segments is allowed. A segmentation must contain at least one segment group.

A sample segmentation is given in Fig. 1. The segmentation contains two segment groups, `A` and `B`, with two segments each. Segments do not overlap and they are labelled according to their segment groups.

The corresponding music sequence is displayed in the pianoroll view (see [6] for details). The figure contains five staff lines and an additional line for Middle C. Notes are displayed as black boxes and the box widths indicate note durations. Lines above notes indicate an increment of the pitch value by one semitone. Rests are represented as grey boxes and vertical lines represent bar changes.

The quality of a segmentation depends on the number and average similarity of its segment groups as well as on the coverage of the segmentation (i.e., the number of notes that are covered by the segmentation). Details about the evaluation of a segmentation are given in the next section.

3. APPLICATION OF A GENETIC ALGORITHM TO THE MUSIC SEGMENTATION PROBLEM

For each instrumental track there exists a high number of potential segment combinations. Since segments can start at any arbitrary position of the composition, the runtime for the evaluation increases exponentially for longer compositions. Therefore, it is not possible to evaluate all potential segmentations but a solution of sufficient quality has to be found in reasonable time. Given these circumstances, the problem domain of music segmentation seems to be highly suited for the application of genetic algorithms.

A music track is encoded as a simple bit vector with one bit for each beat in the music sequence. All bits of value 1 result in new segments starting at the corresponding beats and, therefore, define the segment boundaries within the segmentation. However, this representation does not contain any information about relations between segments so some sort of clustering algorithm has to be included in the evaluation function to detect similarities between segments and to combine them into segment groups. This clustering algorithm compares all segments and calculates similarities considering pitch and rhythm information. The resulting segmentation is analyzed for the fitness calculation. Based on the number of identical and similar segments the evaluation function computes an initial fitness value. The higher the similarities between segments, the higher the initial value. To cover a broader range of characteristica the fitness function evaluates a set of musical features that are not related to similarity. The quality of a segmentation increases if segments have similar durations or regular distances between them. Segments starting at the same time as notes or bars also result in a higher fitness value. Segments overlapping notes or long rests decrease the fitness value as well as very short segment durations or low average similarities. Details about the problem representation and the evaluation function can be found in [11, 13].

To produce the inital population segments are created randomly by assigning 0 or 1 to each bit in an individual. Since segments must be of a minimum length, the probability of 0 is higher than the probability of 1. Various probability ratios are used for different individuals leading to a wider variation of segment durations. Average durations range from the number of beats within a bar up to a maximum value that is set as a parameter of the test run. With a low probability the algorithm ignores the maximum parameter and creates individuals with an average segment duration of $length/beatsPerBar$ with $length$ as the duration of the track in beats (= the length of the bit vector in bits) and $beatsPerBar$ as the number of beats within a bar.

To comply with the schema theorem proportionate selection is used for parent selection. For recombination the authors apply single-point crossover, two-point crossover, and uniform crossover together with the mutation operators that are introduced in the following section.

3.1 Mutation operators

The concept of mutation is important for genetic algorithms to keep diversity within the population and to explore new regions in the search space. Since the mutation operator is problem dependent, it must be adapted to the respective problem. This section lists two mutation operators that are applied to the music segmentation problem.

3.1.1 Bitflip Mutation

Bitflip mutation is the most popular mutation operator for bit vectors. It is also applicable to the music segmentation problem. A bit change from 0 to 1 in a segmentation bit vector corresponds to splitting up an existing segment into two new segments. Changing a bit from 1 to 0 merges two segments into one longer segment. As a result, the bitflip mutation operator creates a new segment or destroys an existing one. The number of segments within a segmentation is usually changed by bitflip mutation.

3.1.2 Bitshift Mutation

The application of bitflip mutation to a bit vector representing a segmentation often has a strong effect on segment boundaries. Therefore, the author introduces a new mutation operator to perform slight changes only: bitshift mutation. The bitshift mutation operator randomly chooses a

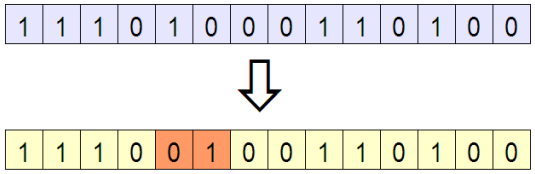

Figure 2: Bitshift mutation

bit of value 1 and shifts it by one bit (see Fig. 2). As a result, the segment boundary is shifted by one beat. The direction of the shift is chosen randomly except for the first and last bit where only one direction is possible. In contrast to the bitflip mutation, the number of segments within the segmentation does not change.

4. THE SCHEMA THEOREM AND THE MUSIC SEGMENTATION PROBLEM

A schema (or similarity template) is built like an individual of the population enhanced with an additional wildcard symbol * which represents *don't care*. The binary alphabet 0,1—the standard alphabet for bit vectors and quite common for genetic algorithms—is extended to 0,1,*. **1**0** is a sample schema for a population of bit vectors with length 6. All bit vectors in the population that have 1 on the 2nd and 0 on the 5th position match the pattern of the schema and, therefore, belong to the sample schema.

Schemata can be described by their defining length and their order. The defining length $\delta(H)$ gives the distance between the first and last fixed position in the schema. Fixed positions are positions that hold values $<> *$. For the sample schema **1**0** the defining length is $\delta(H) = 3$. The order of the schema $o(H)$ is defined by the number of fixed positions which equals the total length minus the number of wildcard symbols. The order of the sample schema is $o(H) = 2$.

A schema represents a region of the search space. Developing various bit vectors of the same schema means exploring a region of the search space in detail whereas the consideration of various schemata leads the genetic algorithm into different regions of the search space. For an individual of length l and an alphabet of cardinality k there are $(k+1)^l$ possible schemata. Each individual of the population is a representative of k^l schemata. A population of n individuals contains at most $n * k^l$ different schemata. During the evolution of the population schemata with higher fitness values will increase whereas weak schemata will disappear. According to Holland's schema theorem the lower bound of the expected number $m(H, t+1)$ of individuals belonging to schema H in generation $t+1$ is given with

$$m(H,t+1) \geq$$
$$m(H,t) * \frac{f(H)}{\overline{f}} * \left[1 - p_c * \frac{\delta(H)}{l-1} - o(H) * p_m\right] \qquad (1)$$

where $f(H)$ is the average fitness of schema H (i.e., the average fitness of all individuals belonging to the schema), $\overline{f}$ is the average fitness of the population (i.e., the average fitness of all individuals in the population), $p_c * \frac{\delta(H)}{l-1}$ gives the probability that H is destroyed by crossover with a crossover

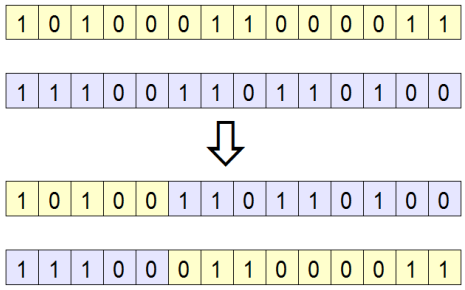

Figure 3: Single-point crossover preserving a schema

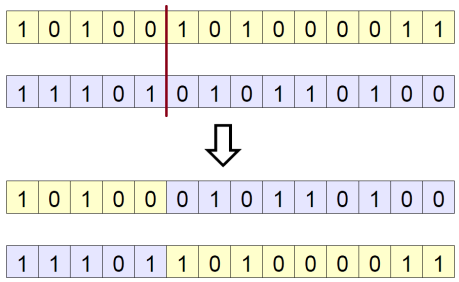

Figure 4: Single-point crossover creating a schema

rate of p_c, and $o(H) * p_m$ gives the probability that H is destroyed by mutation with a mutation rate of p_m.

The traditional schema theorem focuses on a canonical genetic algorithm which applies a binary alphabet, fixed length individuals of equal length, fitness-proportional selection, single-point crossover, and gene-wise mutation. Furthermore, it represents the worst case only. It does not consider that schemata can be preserved in spite of a crossover point within the schema when both parents match parts of the schema, or that children can even match a schema although the parents did not.

Let us consider schema **10001******. Fig. 3 gives an example for the first case. Although the crossover point lies within the schema boundaries and the second parent does not match the schema, it is still not destroyed by the crossover. At least on of the children still matches the schema.

An example for the second case is given in Fig. 4. None of the parents matches the schema, but after crossover at least one of the children does match and, therefore, a new individual of the schema is produced. Neither schema preservation nor schema creation are considered by the original schema theorem, therefore an adaption of the theorem is necessary.

The limitations of the schema theorem mentioned above have been widely criticized and much work has been done to reformulate the theorem. An exact schema theorem has been introduced by several researchers such as Stephens and Waelbrock [15, 16, 17], Whitley [18], Wright [20], Altenberg [1], and Poli [8, 9, 10]. There are two main approaches basing either on strings (microscopic, fine grained) or on schemata (macroscopic, coarse grained).

On the microscopic level Whitley [18] introduced the concept of string losses and string gains for a genetic algorithm using crossover but no mutation. String losses represent strings s that are crossed with other strings and cannot pre-

serve the original string s. String gains occur when two strings that differ from s are crossed and produce s as offspring. Based on this concept the fraction of the population matching schema H at time $t+1$ is defined as

$$P(H,t+1) = p_H * (1 - p_c * losses) + p_c * gains \quad (2)$$

with $p_H = P(H,t) * \frac{f(H)}{\overline{f}}$. This gives an exact value instead of a lower bound only. Furthermore, *losses* and *gains* are not limited to the canonical genetic algorithm but can be defined for various crossover operators. For a genetic algorithm with crossover and mutation the theorem can be extended to

$$P(H,t+1) = p_H * (1 - p_c * ls_c - p_m * ls_m + p_c * ls_c * p_m * ls_m) + p_c * gn_c + p_m * gn_m \quad (3)$$

with ls_c and ls_m for the string losses occuring through crossover and mutation, respectively, and gn_c and gn_m for the string gains resulting from crossover and mutation, respectively.

The authors have chosen Whitley's approach since the schemata for the music segmentation problem can be expressed on a microscopic level. A segment of size $l(s)$ is always defined by a schema of $l(s)+1$ bits. The extra bit is necessary to lock the end of the segment. A segment must start with 1 followed by a series of 0 and concluded by 1. The last 1 does not belong to the segment itself but represents the start of the next segment. Nevertheless it belongs to the schema since it is necessary to define the segment. Otherwise the segment might be turned into a longer segment if followed by 0. All bits within the segment must be 0 since a 1 would end the current segment and start a new one. As a result, there are no wildcards within the definition of a schema representing a segment. To give an example, all segments of size 4 are expressed as schema 10001. Embedded in a track a schema is preceded and/or followed by wildcards. As an example the schema for a segment of size 4 ($sl = 4+1 = 5$) starting at beat 8 of a track of 16 beats ($L = 16$) is defined as ********10001***. Consequently, a schema is defined by a coherent string surrounded by wildcards so the concept of string losses and gains can be applied.

p_c can be omitted from the equation since crossover is always applied. String losses are considered only for the case that the second parent does not match the schema (since a second parent that matches the schema always preserves it). String gains only occur if both parents do not match the schema. The fraction of the population that does not match the schema can be defined as $p_{\overline{H}} = P(\overline{H},t) * \frac{f(\overline{H})}{\overline{f}}$. Since $\overline{f} = P(H,t) * f(H) + P(\overline{H},t) * f(\overline{H})$ and $P(\overline{H},t) = 1 - P(H,t)$ this can be transformed to $p_{\overline{H}} = \frac{\overline{f} - P(H,t) * f(H)}{\overline{f}}$. For simplification we assume that strings not matching the schema are equally distributed among $p_{\overline{H}}$. Whitely included this probability in his *losses* and *gains* calculations. Since the probability to choose a string not matching the schema is independent from the crossover and mutation operators, the authors decided to include it in the general equation instead of the *losses* and *gains* calculations. As a result, the equation is rewritten as

$$P(H,t+1) = p_H * (1 - p_{\overline{H}} * ls_c - p_m * ls_m + p_{\overline{H}} * ls_c * p_m * ls_m) + p_{\overline{H}}^2 * gn_c + p_m * p_{\overline{H}} * gn_m \quad (4)$$

The following sections define losses and gains for various crossover and mutation operators for the music segmentation problem.

4.1 Crossover Operators

The first conditions of a canonical genetic algorithm hold for the genetic algorithm which is applied to the music segmentation problem. Individuals are decoded using a binary alphabet and all individuals representing segmentations of the same track have the same fixed length (which is defined by the duration of the track in beats). The other conditions are met by some of the test settings only. For each crossover and mutation operator string losses and gains must be computed separately.

4.1.1 Uniform Crossover

Uniform crossover combines two parents according to a crossover mask which defines the alleles that should be taken from the first or second parent, respectively. For a track of size L there are 2^L possible crossover masks. However, for a segment of size $sl-1$ only 2^{sl} of them are relevant since it does not matter what happens outside the defined segment area of the schema.

A schema is preserved by uniform crossover if the second parent matches the schema in all positions where there is a 1 in the crossover mask. The same is true for all positions with a 0 in the mask. For a mask with x bits of value 1 there are 2^x possibilities for the first case and 2^{sl-x} possibilities for the second case. Since both cases include the schema itself, there are $2^x + 2^{sl-x} - 1$ possible strings that do not destroy the schema when uniform crossover is applied. For a a crossover mask with x bits of value 1 there will be $2^{sl} - (2^x + 2^{sl-x} - 1)$ out of a total number of $2^{sl} - 1$ strings that destroy the schema (the schema itself is excluded). The probability for each crossover mask to be selected is $1/2^{sl}$. For $0 < x < sl$ there are $\binom{sl}{x}$ crossover masks to be constructed. As a result, the probability to choose a crossover mask with x bits of value 1 is defined by $\frac{\binom{sl}{x}}{2^{sl}}$. The sum over all values of x gives the total losses for the uniform crossover:

$$losses_c = \sum_{x=1}^{sl-1} \frac{(2^{sl} - 2^x - 2^{sl-x} + 1)}{2^{sl} - 1} * \frac{\binom{sl}{x}}{2^{sl}} \quad (5)$$

Two strings not matching the schema can produce offspring matching the schema if one parent matches the schema at all positions that have value 1 in the mask and the other parent matches all positions that have value 0 in the mask. For a mask with x bits of value 1 there are 2^x strings for the first case and 2^{sl-x} for the second. Since the schema itself is excluded from both cases, there are $(2^x - 1) * (2^{sl-x} - 1) = 2^{sl} - 2^x - 2^{sl-x} + 1$ possible string combinations out of a total string combinations of $(2^{sl} - 1)^2$. The sum over all values of x gives the total gains for the uniform crossover:

$$gains_c = \sum_{x=1}^{sl-1} \frac{(2^{sl} - 2^x - 2^{sl-x} + 1)}{(2^{sl} - 1)^2} * \frac{\binom{sl}{x}}{2^{sl}} \quad (6)$$

4.1.2 Single-Point Crossover

Single-point crossover can be expressed as special cases of uniform crossover. For single-point crossover the crossover mask must consist of a coherent set of values 1 and a coherent set of values 0. As a result, there is only one valid mask for each x. String losses can be calculated the same way as for uniform crossover, only the probability to get the specific crossover mask changes. For single-point crossover there are $L-1$ valid crossover masks for the whole track. The crossover mask for the schema is a submask of the whole mask. If the submask contains only 0s or only 1s then the crossover point lies outside the schema boundaries. Those masks do not result in string losses so they are not considered. For a schema of length sl there are exact $sl-1$ valid crossover masks. The sum over all values of x gives the total losses for the single-point crossover:

$$losses_c = \sum_{x=1}^{sl-1} \frac{(2^{sl} - 2^x - 2^{sl-x} + 1)}{2^{sl} - 1} * \frac{1}{L-1} \tag{7}$$

String gains meet the same conditions for the crossover mask. Therefore, string gains for single-point crossover are calculated as

$$gains_c = \sum_{x=1}^{sl-1} \frac{(2^{sl} - 2^x - 2^{sl-x} + 1)}{(2^{sl} - 1)^2} * \frac{1}{L-1} \tag{8}$$

4.1.3 Two-Point Crossover

Like single-point crossover also two-point crossover is a special case of uniform crossover. A crossover mask for two-point crossover must consist of a coherent set of 1s followed by a coherent set of 0s and concluded by a coherent set of 1s. Each set must contain at least one bit. As a result, there are $x-1$ masks for x bits of value 1. For a track of length L there are $\sum_{x=2}^{L-1} x - 1 = \frac{(L-2)(1+(L-2))}{2} = \frac{L^2-3*L+2}{2}$ valid masks. From the schema's point of view there are different valid positions for the crossover points. If both points are outside the schema, the submask contains only bits of value 0 or 1, respectively. If both points hit the schema, the submask has the structure described above. If only one crossover point lies between schema boundaries, the submask is similar to single-point crossover. However, the probabilities change since either the first or the second crossover point can hit the schema. The probability for one point to hit the schema is $\frac{sl-1}{L-1}$ so the probability for each possible point within the schema to be hit is $\frac{1}{L-1}$. The probability for a crossover point to be outside the schema boundaries is $\frac{L-sl}{L-1}$. Since the second point cannot be on the same position as the first one, it changes to $\frac{L-sl}{L-2}$. Either the first or the second crossover point can be inside the schema boundaries, so the probability that exactly one point hits the schema is $\frac{2*(sl-1)*(L-sl)}{(L-1)*(L-2)}$. This results in a probability of $\frac{2*(L-sl)}{(L-1)*(L-2)}$ for each crossover mask with x bits of value 1 if exactly one point falls within the schema boundaries.

For the case that only one point lies inside the schema boundaries the losses are defined as

$$losses1_c = \sum_{x=2}^{sl-1} \frac{(2^{sl} - 2^x - 2^{sl-x} + 1)}{2^{sl} - 1} * \frac{2*(L-sl)}{L^2 - 3*L + 2} \tag{9}$$

and for the case that both points hit the schema as

$$losses2_c = \sum_{x=2}^{sl-1} \frac{(2^{sl} - 2^x - 2^{sl-x} + 1)}{2^{sl} - 1} * \frac{2*(x-1)}{L^2 - 3*L + 2} \tag{10}$$

so the total losses for two-point crossover are defined as

$$losses_c = losses1_c + losses2_c. \tag{11}$$

String gains can be calculated the same way:

$$gains_c = gains1_c + gains2_c \tag{12}$$

with

$$gains1_c = \sum_{x=2}^{sl-1} \frac{(2^{sl} - 2^x - 2^{sl-x} + 1)}{(2^{sl} - 1)^2} * \frac{2*(L-sl)}{L^2 - 3*L + 2} \tag{13}$$

for the case that only one point lies inside the schema boundaries and

$$gains2_c = \sum_{x=2}^{sl-1} \frac{(2^{sl} - 2^x - 2^{sl-x} + 1)}{(2^{sl} - 1)^2} * \frac{2*(x-1)}{L^2 - 3*L + 2} \tag{14}$$

for the case that both points hit the schema.

4.2 Mutation Operators

The schema theorem for a canonical genetic algorithm only holds for gene-wise mutation. Some of the mutation operators applied, however, are not gene-wise, so the schema theorem must be adapted for them.

4.2.1 Bitflip Mutation

The bitflip mutation operator applied to the test runs changes exactly one bit of the whole individual. As a consequence, the probability for each bit to be changed is $1/L$ and the probability that a schema is destroyed by mutation is defined as

$$losses_m = \frac{sl}{L}. \tag{15}$$

New instances of the schema can be created from all strings with Hamming distance 1 to the schema string if exactly the differing bit is mutated. There are sl out of 2^{sl} strings with the required Hamming distance, so string gains for bitflip mutation are defined as

$$gains_m = \frac{sl}{2^{sl}} * \frac{1}{L} = \frac{sl}{2^{sl} * L}. \tag{16}$$

4.2.2 *Bitshift Mutation*

The bitshift mutation operator shifts a 1 from the individual by one position. As a consequence, the probability of a schema to be destroyed is independent from its length since every schema contains two bits of value 1 (one to start and one to end the segment). The only factor that influences the probability to destroy a schema is the number of bits of value 1 within the individual. Each individual is created based on a 0:1 ratio z (with $z > 1$) which results in an average number of $L/(z+1)$ bits of value 1. A schema is always destroyed if the chosen 1 is shifted to the inside ($p = 1/2$). If it is shifted to the outside and the outer bit does not have value 1 ($p = 1/2 * 1/2 = 1/4$) it is also destroyed. An outer value of 1 preserves the schema since an exchange of two bits of value 1 does not change the string. As a result, the losses for bitshift mutation are defined as

$$losses_m = \frac{2*(z+1)}{L} * \frac{3}{4} \tag{17}$$

where z in a best case scenario is defined by the length of the schema so that $z + 1 = ls$.

A shift to the right of a bit of value 1 creates a new instance of the schema if this bit is either the start of an extended schema string with an additional 0 (e.g., 100001 for schema 10001) or if it is the end of a shortened schema string with one 0 missing followed by 0 (e.g., 10010 for schema 10001). The same holds for a shift to the left by exchanging the two cases. For the first case, there is one out of 2^{sl+1} strings matching the required pattern since the extended schema is defined by $sl + 1$ positions. For the second case there is one matching string out of 2^{sl} since $sl - 1$ positions define the shortened pattern plus one position for the following 0. As a result, the gains for bitshift mutation are defined as

$$gains_m = \frac{2*(z+1)}{L} * (\frac{1}{2^{sl+1}} + \frac{1}{2^{sl}}). \tag{18}$$

If a bit of value 1 is the first or last bit of a bit vector, it can be shifted in one direction only. For simplification reasons this case has been omitted since it would only give a slight change to the equation and can be neglected for longer L.

4.3 Schema Theorem for a Concrete Music Segmentation Problem

A very well known children's song, *Frere Jacques*, has been chosen as a sample track. Fig. 5 displays the song's notes, the solution to the segmentation problem for this song, and the bit vector representing this solution.

The sample track has a duration of 32 beats, so $L = 32$. sl is defined by the duration of the segments which is 4 beats. As a result, $sl = 5$. This does not hold for the last segment since it is only defined by 4 bits plus the end of the track, but this exception will be ignored.

10001000100010001000100010001000 is the bit vector representing the best solution containing all segments from Fig. 5. The schemata defining the segments of the optimal solution are given in Table 1 and comply with the building block hypothesis formulated by Goldberg [2]: *short, low level schemata (i.e., schemata of low defining length and low order) with high fitness values have a higher chance to survive the evolution process. Those building blocks are combined during the evolution of generations to form longer, highly fit blocks which, in the end, result in the optimal solution.*

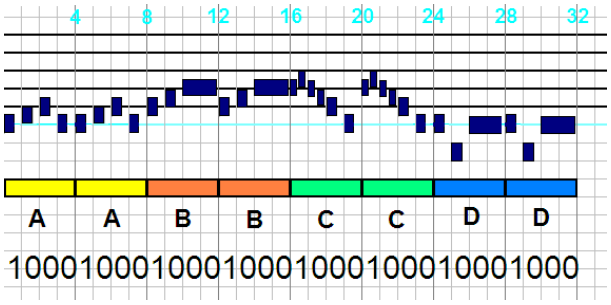

Figure 5: Segmentation for Frere Jacques

4.4 Concrete Values for Various Crossover and Mutation Operators

The previous sections defined string losses and gains for various crossover and mutation operators. The following sections will compute concrete values for the given music track.

4.4.1 *Uniform Crossover*

$$losses_c = \sum_{x=1}^{4} \frac{(33 - 2^x - 2^{5-x})}{31} * \frac{\binom{5}{x}}{32} = 0.5746 \tag{19}$$

$$gains_c = \sum_{x=1}^{4} \frac{(33 - 2^x - 2^{5-x})}{31^2} * \frac{\binom{5}{x}}{32} = 0.01854 \tag{20}$$

4.4.2 *Single-Point Crossover*

$$losses_c = \sum_{x=1}^{4} \frac{(33 - 2^x - 2^{5-x})}{31} * \frac{1}{31} = 0.07492 \tag{21}$$

$$gains_c = \sum_{x=1}^{4} \frac{(33 - 2^x - 2^{5-x})}{(31)^2} * \frac{1}{31} = 0.00242 \tag{22}$$

4.4.3 *Two-Point Crossover*

$$losses_c = \sum_{x=2}^{4} \frac{(33-2^x-2^{5-x})}{31} * \frac{54}{930} + \sum_{x=2}^{4} \frac{(33-2^x-2^{5-x})}{31} * \frac{(x-1)*2}{930} = 0.11426 \tag{23}$$

$$gains_c = \sum_{x=2}^{4} \frac{(33-2^x-2^{sl-x})}{31^2} * \frac{54}{930} + \sum_{x=2}^{4} \frac{(33-2^x-2^{sl-x})}{31^2} * \frac{(x-1)*2}{930} = 0.00369 \tag{24}$$

Although uniform crossover is more disruptive than single-point or two-point crossover (see [19]), there are other advantages of this operator. Spears and DeJong [14] stated that "With small populations, more disruptive crossover operators such as uniform or n-point ($n >> 2$) may yield better

Table 1: Schemata for Segments of the Best Solution

Segment	Schema name	Schema
A (first occurence)	H_1	*10001***********************
A (second occurence)	H_2	****10001*********************
B (first occurence)	H_3	********10001******************
B (second occurence)	H_4	************10001**************
C (first occurence)	H_5	****************10001**********
C (second occurence)	H_6	********************10001*******
D (first occurence)	H_7	************************10001***
D (second occurence)	H_8	****************************1000

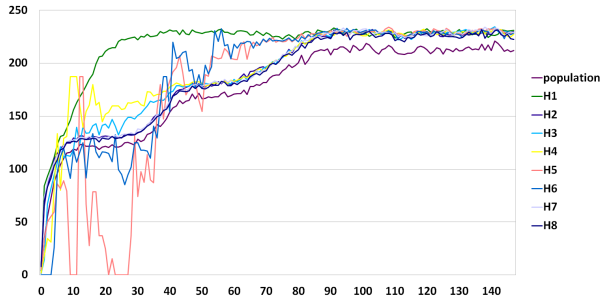

Figure 6: Average fitness of individuals matching one of the schemata

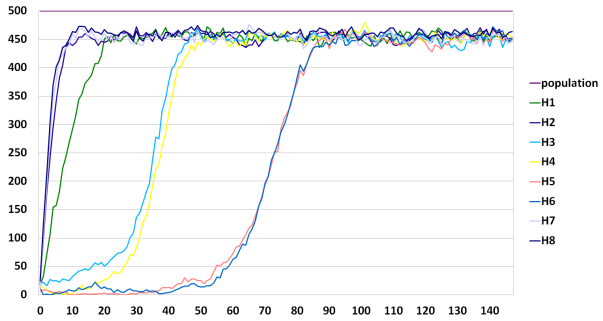

Figure 7: Average count of individuals matching one of the schemata

results because they help overcome the limited information capacity of smaller populations and the tendency for more homogeneity." In small populations uniform crossover can help to avoid premature stagnation in local maxima.

4.4.4 *Bitflip Mutation*

$$losses_m = \frac{5}{32} = 0.15625 \tag{25}$$

$$gains_m = \frac{5}{32} * \frac{1}{32} = \frac{5}{1024} = 0.00488 \tag{26}$$

4.4.5 *Bitshift Mutation*

$$losses_m = \frac{8}{32} * \frac{3}{4} = 0.1875 \tag{27}$$

with a ratio of 3:1 for bits of value 0 : bits of value 1 so $z = 3$.

$$gains_m = \frac{8}{32} * (\frac{1}{64} + \frac{1}{32}) = 0.01172 \tag{28}$$

4.5 Development of Schemata for the Concrete Music Sequence

This section examines the generation-wise development of individuals matching the schemata given in Table 1 for single-point crossover and bitflip mutation. Schema names are used according to Table 1.

Fig. 6 displays the average fitness of each schema as well as the average fitness of the whole population during the first 150 generations. The algorithm was run with 1000 generations but values do not change significantly in later generations. With the exception of some schemata during the first 50 generations, the fitness of individuals matching a schema is always higher than the average fitness of the whole population. Corresponding to the building block hypothesis the number of individuals matching the schemata increases during the evolution process. This process is illustrated in Fig. 7 which shows the average count of individuals matching a schema. The number of individuals for some schemata increases faster than others, but after generation 90 all schemata have an individual count of around 450 (out of 500 individuals in the whole population).

The previous sections presented an adapted version of the schema theorem to compute the expected numbers of individuals in a population that match a schema. To conclude this section the observed numbers of schema matching individuals are compared to the expected values. Schema H_3 is chosen as a sample schema. Fig. 8 gives the expected and observed development of the number of individuals in a population of 500 matching schema H_3. Small differences stem from the asumption in the equation that the population is infinite. However, the observed numbers almost match the expected ones, so the equations presented in this paper give a good approximation for the development of schemata during a genetic algorithm.

5. CONCLUSIONS

Genetic algorithms can successfully be applied to the music segmentation problem. This paper described the theoretical background for their success by adapting the well known schema theorem to this specific problem. The reader was introduced to the music segmentation problem as well as to the theory behind the application of genetic algorithms to this problem.

A practical example illustrated some schemata for a given music sequence and the development of individuals matching them during the evolution process. The results are promising since they show that the expected numbers of individuals

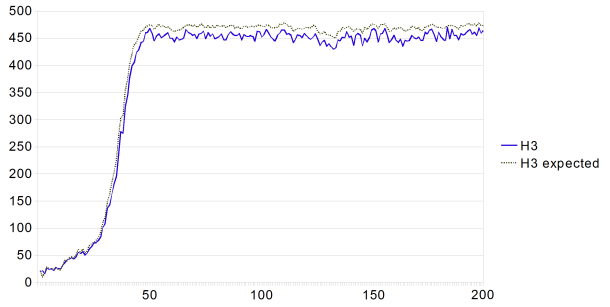

Figure 8: Expected and real number of individuals matching schema H3

are generally met. Furthermore, the building block hypothesis holds for the sample music sequence. Short segments of above-average fitness survive the evolution process and, in the end, result in the optimal solution.

Future work will include more test runs for other combinations between mutation and crossover to evaluate if the expected numbers of schema matching individuals given in this paper are also met for those combinations.

6. REFERENCES

[1] L. Altenberg. The schema theorem and price's theorem. In *Foundations of Genetic Algorithms 3*, pages 23–49. Morgan Kaufmann, 1995.

[2] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison Wesley Longman, 1989.

[3] C. Grilo and A. Cardoso. Musical pattern extraction using genetic algorithms. *Computer Music Modeling and Retrieval*, 2771:124–154, 2004.

[4] J. H. Holland. *Adaptation in Natural and Artificial Systems*. Ann Arbor: The University of Michigan Press, 1975.

[5] B. S. Ong. *Structural Analysis and Segmentation of Musical Signals*. PhD thesis, Universitat Pompeu Fabrat, 2006.

[6] N. Orio. *Music Retrieval: A Tutorial and Review*. Now Publishers Inc, 2006.

[7] N. Orio and G. Neve. Experiments on segmentation techniques for music documents indexing. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, 2005.

[8] R. Poli. Exact schema theorem and effective fitness for gp with one-point crossover. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 469–476. Morgan Kaufmann, 2000.

[9] R. Poli. Hyperschema theory for gp with one-point crossover, building blocks, and some new results in ga theory. In *Genetic Programming Proceedings of EuroGP2000*, pages 163–180. Springer-Verlag, 2000.

[10] R. Poli. Recursive conditional schema theorem, convergence and population sizing in genetic algorithms. In *Proceedings of the Foundations of Genetic Algorithms Workshop (FOGA 6*, pages 143–163. Morgan Kaufmann, 2000.

[11] B. Rafael and S. Oertl. A two-layer approach for multi-track segmentation of symbolic music. In M. H. Hamza, editor, *Artificial Intelligence and Applications*, pages 157–164. ACTA Press, 2010.

[12] B. Rafael, S. Oertl, M. Affenzeller, and S. Wagner. Music segmentation with genetic algorithms. In *Twentieth International Workshop on Database and Expert Systems Applications*, pages 256–260, 2009.

[13] B. Rafael, S. Oertl, M. Affenzeller, and S. Wagner. Using heuristic optimization for segmentation of symbolic music. In *Computer Aided Systems Theory - EUROCAST 2009*, pages 641–648, 2009.

[14] W. Spears and K. DeJong. An analysis of multi-point crossover. In *Foundations of Genetic Algorithms*. G. Rawlins, 1991.

[15] C. Stephens. Some exact results from a coarse grained formulation of genetic dynamics. In *Proceedings of GECCO 2001*, pages 631–638. Morgan Kaufmann, 2001.

[16] C. Stephens and H. Waelbroeck. Schemata evolution and building blocks. *Evolutionary Computation*, 7:109–124, 1998.

[17] C. R. Stephens and H. Waelbroeck. Effective degrees of freedom in genetic algorithms. *Phys. Rev. E*, 57(3):3251Ű3264, 1998.

[18] D. Whitley. An executable model of a simple genetic algorithm. In *Foundations of Genetic Algorithms 2*, pages 45–62. Morgan Kaufmann, 1992.

[19] D. Whitley. A genetic algorithm tutorial. Technical report, Colorado State University, 1993.

[20] A. H. Wright. The exact schema theorem. Technical report, University of Montana, 1999.