Memetic Optimization of Video Chain Designs

Walid Ali [1] and Alexander Topchy [2]

[1] Philips Research Laboratories, 345 Scarborough Rd, Briarcliff Manor, NY, 10510, USA
walid.ali@philips.com

[2] Department of Computer Science and Engineering
Michigan State University, East Lansing, MI, 48824, USA
topchyal@cse.msu.edu

Abstract. Improving image quality is the backbone of highly competitive display industry. Contemporary video processing system design is a challenging optimization problem. Generally, several video algorithms must be sequentially applied to real-time video data. Overall image quality depends on the nonlinear interactions between multiple design parameters: variable settings for each module (algorithm), the amount of data being transferred in the video processing chain as well as the order of the cascading modules. Unfortunately, no systematic techniques are currently available to configure the video chain without lengthy trial and error process. We propose a rapid and reliable method for optimization of composite video processing systems based on genetic algorithm coupled with local search heuristics. Video system configuration is evolved toward the best image quality, driven by an objective video quality metric. We analyze several local search approaches, including hill-climbing, simplex and estimation-of-distribution algorithms. Experimental study demonstrates superior performance of memetic strategies over the conventional genetic algorithm. We obtain novel and practical video chain solutions that are typically not attainable by regular design process.

1 Introduction

Picture quality is a key factor influencing consumer brand name preference for video communication devices [1, 25, 27]. These appliances range from small hand-held devices to a large complicated television sets. They all deploy a number of video processing modules, which interact together to create the desired output picture. Generally, video algorithms are developed and evaluated in isolation from the actual video processing system, of which they will be a part in a consumer product. Obviously, the final quality depends on the interaction of the constituent algorithms. This interaction depends on the order in which these modules are applied, as well as on the settings of each algorithm's programmable parameters. Whereas the development of individual video processing algorithms (modules) involves a lot of analysis and simulation, the development of larger chains often involves a more ad-hoc approach. However, a thorough analysis of the inter-algorithm interaction is required in order to

K. Deb et al. (Eds.): GECCO 2004, LNCS 3103, pp. 869–882, 2004.

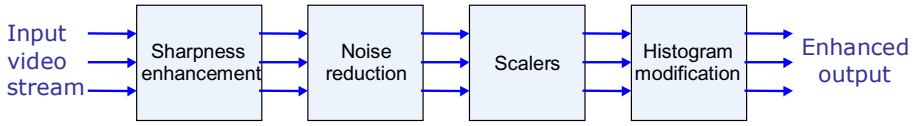

Fig. 1. An example of video chain configuration with four modules (algorithms).

find the optimal system architectural design and the best tuning of each individual module's parameters.

This challenge can be defined as a formal optimization problem called Video Chain Optimization (VCO) [24]. A genetic algorithms approach has been proposed to drive the optimization process [4, 7, 10, 24] and has proven to be successful in leading the search toward the global optima. However, this method is hindered by the time it requires to converge to satisfactory design solutions. Simulation time on order of days is usually necessary to reach the appropriate designs. This is mainly due to the computational complexity of video algorithms and the need to simulate a large number of video systems before converging to (sub)optimal configurations and settings of video chain.

This paper presents our approach for speeding up both the convergence rate and the execution time of optimization. We speed up the convergence by augmenting gray-coded genetic algorithm (GA) with a less complicated search method (local search). Thus, we benefit from the GA's ability to traverse the overall search space without being trapped in local optima while converging faster using neighborhood move operators. Furthermore, the execution speed is improved by keeping the samples visited over the search path in memory and using them for future search trials.

2 Background on Video System Design

The video-processing filter to be optimized consists of multiple video-processing modules, which are considered essential for high-end and top-end television sets. We deal primarily with video signals in the YUV and RGB domains, i.e. with image enhancement and display adaptation functions [9, 14]. Tuning, IF/color decoding, and channel/source decoding are not considered for now. The functions used are luminance peaking by sharpness enhancement, spatial scaling, noise reduction and histogram modification [9]. For example, Fig. 1 shows a particular video chain configuration with these functions assigned to separate modules.

Sharpness enhancement, which nowadays is a common feature in TV sets, focuses on improving the perceived sharpness of the luminance signal. Boosting the higher frequencies in the luminance signal can enhance the sharpness. The *noise reduction* unit reduces the higher frequency components based on measuring the presence of noise. The *scalers* are implemented using polyphase FIR filters. The horizontal scalers process each line of input video data and generate a horizontally scaled line of output video data. In the case of expansion, this is done by up-sampling that is per

order of functions	parameters of function F_1	parameters of function F_2	…	parameters of function F_n

Fig. 2. A general structure for the chromosome representing a video processing chain

formed either by a polyphase filter for which the horizontal expansion factor determines the filter phases required to generate each output pixel, or by a filter that uses this factor to interpolate the output pixels from the input pixels. In the case of compression, a transposed polyphase filter is used to down-sample the input data, and the horizontal compression factor determines the required filter phases. The vertical scalers, however, generate a different number of output video lines than were input to the module, with input and output lines having the same numbers of pixels. Vertical scalers may either compress or expand the number of output video lines. *Histogram modification* stretches out the luminance value for the black color and the white color to better represent the color contents of the video sequence.

3 Applying Genetic Algorithms to the Video Chain Design

The optimization process utilizes genetic algorithms (GA) to search for the parameter settings, implementation alternatives and an interconnection scheme delivering the best objective picture quality. In optimizing the video-processing scheme, a chromosome defines a certain way in which different video processing modules are connected and video stream is transformed. A chromosome consists of a number of genes. The genes in the video optimization problem encode the parameters of video processing functions as well as the order of modules, which determines the connection scheme. Fig. 2 shows a general structure of the string (chromosome) encoding a video processing chain with n functions.

We optimized a video processing system, which consisted of four cascaded video processing modules, namely, a spatial poly-phase scalar, a noise reducer, a sharpness enhancer and a histogram modification module. The optimization algorithm deals with each module as generically as possible. It assumes no prior information about the modules nor the connectivity constraints on the cascaded modules. The search process seeks not only the optimal values of pre-defined set of parameters within each module, but also the data bus parameters. The data precision (number of bits in a data bus, i.e., bus width) between two cascaded modules is considered a parameter to be optimized. We elected to use this set of video processing modules because of their vital role in any video system [9, 23]. Moreover, some of these modules are competing modules [9, 14], e.g., increasing the sharpness would enhance the perceived existing noise and reducing the noise will blur the picture resulting in the loss of appealing crispiness.

The complete optimization scheme consists of three main components: (i) model of the video processing system, (ii) the objective image quality measurement component, and (iii) the search procedure with genetic algorithm in its core. It must be noted

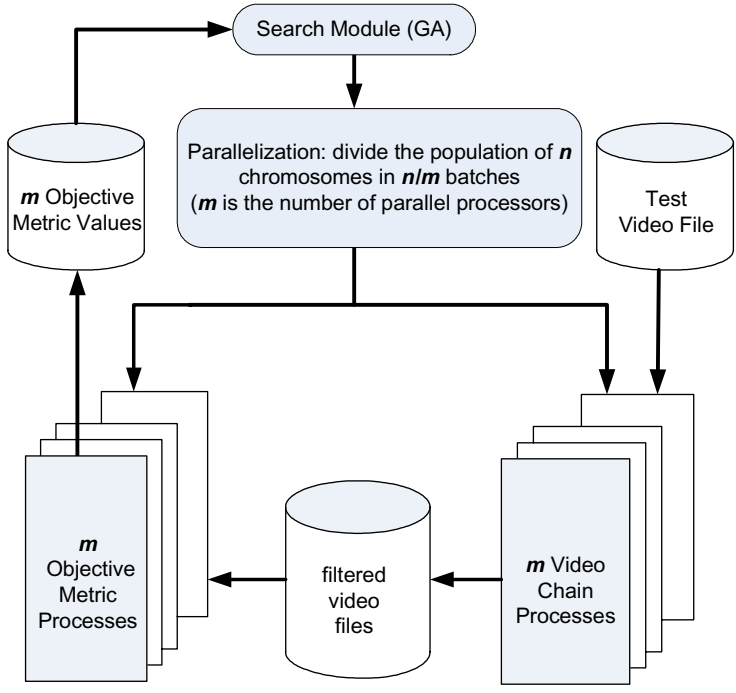

Fig. 3. A schematic diagram for optimizing a video system

function order 5 bits	Noise Reduction 2 bits	sharpness 3 bits	Data Bus Width 3 bits	Data Bus Width 3 bits	Data Bus Width 3 bits

Fig. 4. The chromosome encoding for a constraint-free video chain

that the video system is entirely modeled in software. However, the obtained design solutions are easily transferred to the ultimate display product based on specialized hardware.

The computational bottleneck in this scheme results from the simulation complexity of the video processing system. We run a number of video processing systems in parallel (depending on the available computation processors on a parallel computer), as well as a number of the objective image quality metric components. Parallelizing the computationally demanding portions of the system significantly improves the performance. Fig. 3 shows a schematic diagram of the overall system.

3.1 A Scalable Random Order Video Processing System

The primary goal of this study is the optimization of scalable random order video system. In particular, we consider the video-processing filter without any prior constraints on the modules' parameters, order and the data bus width. The noise reduction unit has a smearing effect, which can be controlled by four settings: 1, 2, 3 or 4, thus 2 bits are needed to represent it. The sharpness enhancement has a parameter with five settings (3 bits). The number of bits transferred between any two cascaded video processing units could range between 8 and 12 bits (5 settings; 3 bits are needed for encoding). There are 24 possible ways of cascading 4 video-processing modules, and 5 bits are needed to represent it. Thus, the chromosome needed to represent this video-processing filter comprises 19 bits. The video filter has 60,000 possible variations. Fig. 4 shows the chromosome structure in this case.

4 Objective Image Quality Metrics as Fitness Function

Evaluation of video quality has always been achieved using subjective methods [1, 11]. Since the subjective results vary according to the variability between the viewing audiences, subjective results, which are solely based on perception, have to be statistically post-processed in order to remove the ambiguity resulting from the non-deterministic nature of these results. Linear and nonlinear heuristic statistical models [21] have been proposed to normalize these results, and produce certain figures of merit to represent the goodness/degradation of the video quality. However, relying on human evaluation is expensive and sometimes impossible to adopt. Apparently, a subjective evaluation cannot be used within the optimization loop, when tens of thousands of trials are performed. Hence, automatic methods to evaluate video quality are necessary.

Ideal automatic and objective assessment of video quality should highly correlate with subjective testing [5, 25]: the higher the correlation, the better the objective method is. In practice, different methods are investigated for objective image quality. They vary widely in complexity and performance and can be categorized in several ways, measuring traditional analog vs. digital artifacts, measuring the general perceptual quality of a video sequence vs. measuring a specific artifact only, and finally still image (frame/field) evaluation vs. temporal evaluation. For instance, nine proponent models were proposed to the Video Quality Expert Group (VQEG) [25].

We employ a composite scalable objective metric, which consists of a set of metrics, each of which is geared toward measuring a certain feature of the video sequence. Each of these n metrics gives a reading, f_i, $(1 \le i \le n)$, which measures a certain feature of the video sequence I. These readings are weighted by a weight factor each, w_i $(1 \le i \le n)$ and linearly combined:

$$F = \sum_{i=1}^{n} w_i f_i(I) \tag{1}$$

Of course, the set of weights $\{w_i\}$ must be determined in advance. Weights are found by maximizing the correlation factor R between human perceptual evaluation and the composite objective measure F on the baseline training data. In particular, we maximized Spearman rank order [21] that measures the correlation factor R_S between the subjective evaluations X_r and the objective metrics $Y_r = F(I_r)$ over a number of training video samples $\{I_r\}$:

$$\max_{w_1,\ldots,w_n} R_S(w_1, w_2, \ldots, w_n) \tag{2}$$

$$R_S = 1 - \frac{6\sum_{r=1}^{m}(X_r - Y_r(w_1, w_2, \ldots, w_n; I_r))^2}{m(m^2-1)} \tag{3}$$

A more detailed explanation and illustration of the objective image quality metric and its constituent metrics is given in [24].

5 Memetic Optimization Approach

The problem we are trying to solve, video chain optimization, has a distinctive set of features. Studying the nature of these features is the best way to decide on the most suitable optimization method to adopt. VCO problem is computationally expensive, since it deals with video processing, has a highly nonlinear, non-differentiable cost function, and is strongly constrained, (e.g., the admissible range of any parameter), and finally, it can be parallelized.

For several decades it has been well understood that competitive optimization methods should include both global and local search as a tradeoff between exploration and exploitation of search space. The evolutionary search is no exception. Indeed, evolutionary search proved to be efficient in exploration of large solution spaces. Genetic algorithms deserve special attention since they are known to be robust across the wide spectrum of optimization problems [8] including real world applications. However, genetic algorithms are not particularly successful in fine-tuning of the candidate solutions. GAs can quickly identify promising search areas. Implicitly operating with many relatively short building blocks, a GA quickly converges toward better solutions. But if only a few bits are not correctly set in the chromosome, it may be very difficult for a GA to find them. An extra search is often necessary for the fine-tuning of the solutions. For solutions that are time-critical, proper local search component plays important role.

Though many hybrid optimization techniques are in use with or without GA playing a part in a search, we focus on the genetic algorithm and local search (LS) combination. Several names are applied to this combination. It is known as particular case of metaheuristic algorithms [28], Lamarckian evolution, Baldwin effect [26], genetic local search [22], or more recently as memetic algorithms [19]. Actual problem dictates the choice of local search component. Many successful applications utilized hill-climbing [16], gradient-descent [15] and Nelder-Mead's Simplex [20]. Extra local

search usually exploits the best solutions within population and moves to new points via small modification of the old solutions. Thus, exploitation of the existing information is achieved by local search in the vicinity of the best solutions. Certainly, the definition of the small moves depends on the metric imposed upon the search space and corresponding encoding of solutions. Typically, an elementary move causes minimal change in the solution under some distance definition. For example, choice of Hamming distance between solutions in binary representation naturally induces a bit flip as elementary modification. The immediate neighborhood of a point consists of all points reachable via a single bit flip. With different representation of solutions neighborhoods may radically differ.

In order to improve optimization speed, a knowledge-based search strategy can be used. Since no prior information on the search space is available, we extract this information from the GA population. Thus, it is important to decide on how to combine GA with local search. There are many ways to achieve this. First, we have to decide on whether the local search interleaves with evolutionary operators during a regular GA run or if that happens only at the end of a GA run as a form of fine-tuning. Taking into account that GA rather quickly provides good indications for best global regions in the search space, it is reasonable to allow limited amount of search in the vicinity of the current best solutions during the GA run.

It is also important to decide if the changes due to the local search are coded back to the chromosome or not, that essentially is the choice between Lamarckian and Baldwin type of evolution [26]. In Lamarckian evolution, learning (via local search) affects fitness distribution as well as the underlying genotype, while the Baldwin effect is mediated via the fitness values only. In our case, the question is whether locally learned values are copied back into the genotype (Lamarckian) or whether the chromosome remains unmodified while the individual's fitness is changed by local search (Baldwin).

Even though there are indications that Lamarckian and Baldwin evolution demonstrate comparable performance in some circumstances [26], our approach is to allow modified genetic code to be written back to the chromosome, thus following Lamarckian style, which proves to better improve the GA solution. Below we will discuss three different implementations of hybrid GA used in our experimental study.

5.1 Next-Ascent Stochastic Hill Climbing

This type of *hill climbing* makes a change in a random gene. If this change results in better or equal fitness value, then a new solution is accepted. It differs from steepest ascent since we do not exhaustively search through the neighborhood for the best improvement, but rather accept *any* improvement. The accepted solution becomes a member of the current population at this moment. In a parallel implementation, several current best solutions are subjected to hill-climbing modifications simultaneously.

5.2 Simplex Method

The Nelder-Mead's simplex algorithm [13] uses a geometric figure consisting of d+1 points in d dimensions. In this method, a new simplex is created from an old simplex by replacing the worst evaluation point with a new point. The previous worst point is reflected over the mean vector $\mathbf{c}$ of d best points to create a new point:

$$\mathbf{c} = \frac{1}{d}(\mathbf{x}_1 + ... + \mathbf{x}_d) \tag{4}$$

$$\mathbf{x}_{\text{new}} = \mathbf{c} + \mathbf{c} - \mathbf{x}_{\text{worst}} \tag{5}$$

The new point is evaluated, and simplex transformation continues as before. In order to create every new point, d+1 points are drawn at random from the current population. Many modifications to this basic procedure are known. The most popular approach [13] uses adaptive expansion and contraction of the simplex depending on the objective function value of a new point. A variant of Nelder-Mead's simplex was also used in hybrid GAs [20, 29, 3]. Our GA-Simplex hybrid iterates GA with limited amount of simplex search every generation.

5.3 Local Search Directed by Estimation Distribution Algorithm

Any knowledge about the problem is potentially valuable because it allows us to bias the search toward more promising solutions. Often this knowledge is gained only during the optimization itself. Therefore, it is important to control the search on the fly by the information that is explicitly accumulated with the experience. Explicit search statistics [2] can be used to shape this information and make it useful for generating new solutions. A whole family of such algorithms relies on learning the probability distributions among the best solutions [17]. New points are sampled from such a distribution, evaluated, and used to refine the distribution, in turn. These methods are known as *estimation of distribution algorithms* (EDA) or *probabilistic modeling* and can be used instead of GA for all optimization purposes. In our approach we do not abandon GA as an exploration tool, but use explicit probability distributions for the local search only. Though many models are suitable for capturing the distribution, we use the simplest and most inexpensive one: marginal distribution of individual alleles. This distribution is simply summarized by the vector of probabilities of alleles at every locus $\mathbf{P}_{\text{good}}=\{P_{\text{good}}(x_1), P_{\text{good}}(x_2),\ldots, P_{\text{good}}(x_n)\}$ assuming that alleles are independent. It approximates the full joint probability distribution of the alleles of the best solutions as the product of unconditional probabilities of individual alleles:

$$P(x_1, x_2,\ldots, x_n) = P_{\text{good}}(x_1)\, P_{\text{good}}(x_2) \ldots P_{\text{good}}(x_n) \tag{6}$$

With binary coding, the components of vector $\mathbf{P}_{\text{good}}$ contain the probabilities of 1's at the corresponding genes. This distribution is used to guide the local search. Every candidate solution subjected to the local search is modified by the distribution $\mathbf{P}_{\text{good}}$.

In contrast to other EDA-type algorithms, we blend the distribution with the candidate individual. Thus, instead of drawing completely new individuals only from the distribution $\mathbf{P}_{good}$, the candidate solution is modified by the information on good solutions available at the current generation. The modified individual is sampled from the distribution $\boldsymbol{Q}(x_1, x_2,\ldots, x_n)$:

$$\boldsymbol{Q}(x_1, x_2,\ldots, x_n) = \left\{ \frac{x_1 + P_{\text{good}}(x_1)}{2}, \frac{x_2 + P_{\text{good}}(x_2)}{2}, \ldots, \frac{x_n + P_{\text{good}}(x_n)}{2} \right\} \tag{7}$$

Sampling from the distribution **Q** produces new solutions closer to the old solution in comparison with simply drawing from the distribution $\mathbf{P}_{good}$. This is more adequate to our goal of making just slight modifications in the vicinity of the old solutions. Unlike mutations, EDA-based modifications are not uniformly random, but rather directed by the currently known distribution of the good solutions. In general, one can apply unequal weights to the shares of probabilities coming from **x** and from $\mathbf{P}_{good}$:

$$\mathbf{Q} = \lambda\mathbf{x} + (1\text{-}\lambda)\mathbf{P}_{good} \tag{8}$$

In addition, we need to take care that the information in $\mathbf{P}_{good}$ is properly updated every generation. In our implementation, only solutions whose fitness value exceeds the mean population fitness contribute to $\mathbf{P}_{good}$. Distribution of good solutions from the *current* population is summarized in $\mathbf{P}_{curr.good}$ at each generation and is used to update the distribution $\mathbf{P}_{good}$ estimated from the previous generations:

$$\mathbf{P}_{good}(t) = \alpha\mathbf{P}_{good}(t\text{-}1) + (1\text{-}\alpha)\mathbf{P}_{curr.good} \tag{9}$$

By altering the factor $0<\alpha<1$ we can reduce the contribution of the previous distribution and introduce new information updating $\mathbf{P}_{good}$ incrementally, from generation to generation.

6 Empirical Study

A real video chain described in section 3.1 was optimized in the experiments. The GA package *dCHC* by Eshelman [6] served as a basis for all hybrid algorithms. We have studied the performance of three different combinations of a genetic algorithm with local search procedures, and compared them to the pure GA solution obtained by dCHC algorithm. The following memetic algorithms were implemented:

1. GA + simplex local search
2. GA + next-ascent stochastic hill-climbing
3. GA + EDA-based local search

The dCHC algorithm was used as the baseline genetic algorithm. Both half uniform (HUX) and 2-point crossover were employed for genetic recombination [6]. Other typical features of dCHC include cross-generation selection, maintaining diversity and soft restart. All the experiments were conducted with population size 50. Variable amounts of 10-20 modifications by local search were performed on the best individu-

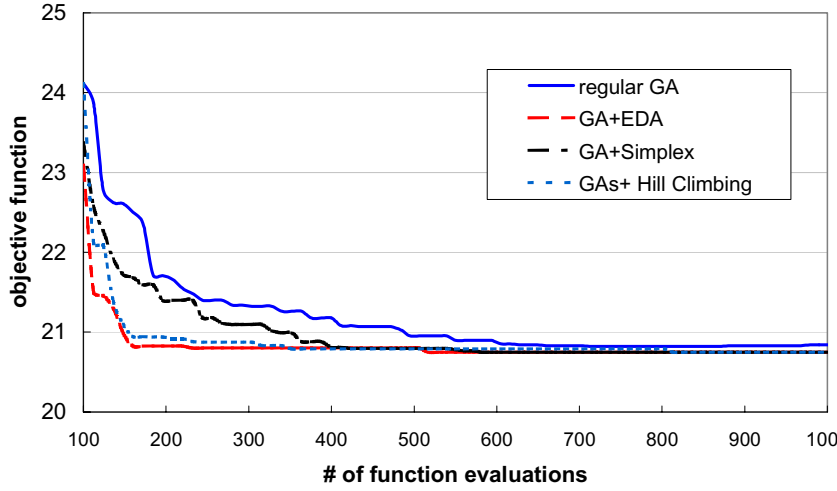

Fig. 5. The best solution found after a set of trials for different memetic heuristics

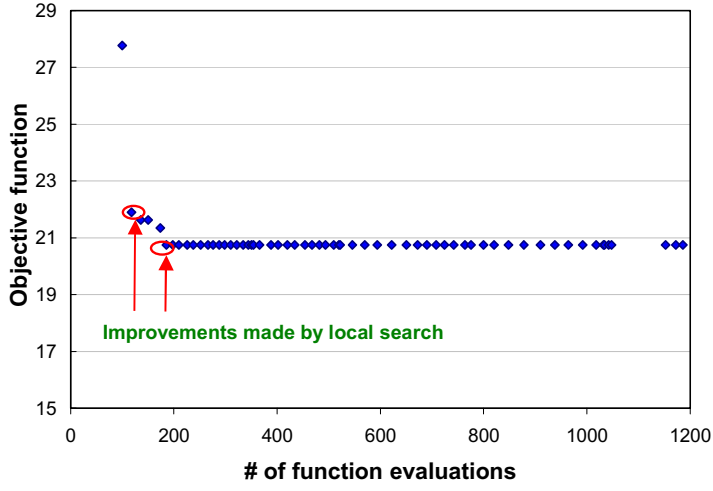

Fig. 6. A typical improvement provided by local search in hybrid GA+EDA algorithm.

als every generation. The number of modifications depends on the number of off spring produced, which varies from generation to generation due to incest prevention mechanism [6]. Since we evaluate the children in parallel, some processors may be free and therefore can be used to run extra evaluations issued by the local search. On average, the relative effort by the local search was lower in the beginning of the run and higher before the convergence, since the number of new children produced by dCHC in every generation is smaller as the population converges. A video chain configuration was represented by a chromosome, which has 6 different genes and 19 total

bits. A high quality solution for this problem was known in advance as found by multiple runs of pure GA, each run taking several days to complete. All the trials, including local search, were done in parallel in batches of 10 evaluations (number of used parallel processors = 10). Two types of information were of particular importance: the average number of trials to reach the best solution and the dependence of the current best performance on the number of evaluations.

Fig.5 shows the objective function value (image metric) of the best solution as a function of the number of trials (objective function evaluations) averaged over 20 different runs. The total count of trials includes both GA trials as well as local search trials. The main observation is that all the hybrid algorithms performed better than the original genetic algorithm. All hybrid algorithms operated with strings in Gray coding representation for the parameters of the video chain. A minor difference in favor of the Gray representation was found in the experiments. The sample VCO problem was solved fastest by the hybrid algorithm with the local search guided by the estimation distribution algorithm. The least improvement was achieved by the GA + simplex hybrid algorithm.

As seen from the Fig. 5, hybrid GAs have converged faster than the pure GA. This is simply because of the local search's better ability to fine-tune the solution faster than a population-driven (like GAs) search method. Fig. 6 shows a typical improvement achieved by introducing local search to pure GAs.

Table 1 summarizes trial statistics to reach the known global optima optimal solution for the video chain optimization problem using different combinations of GA with local search procedures. We report the average and the standard error of the number of trials needed to reach the best solution.

Table 1. Average number of trials necessary to reach the best known VCO solution for different hybrid algorithms, averaged over 30 runs each.

Algorithm	Average	Std. Err.
GA + EDA	**268**	21
GA + Hill-climbing	336	48
GA + Simplex	481	40
GA	635	141

6.1 Improving Execution Time by Solution Caching

Since the most time consuming process is applying the objective image quality metric, we can use a fast retrieval memory mechanism (e.g., a hash table) to store each evaluated chromosome (video system) together with its associated image quality value. Utilizing the memory unit prevents many costly evaluations, which would otherwise unnecessarily slow down the optimization process. Without caching the evaluated configurations, we may need to re-evaluate a chromosome if redundant

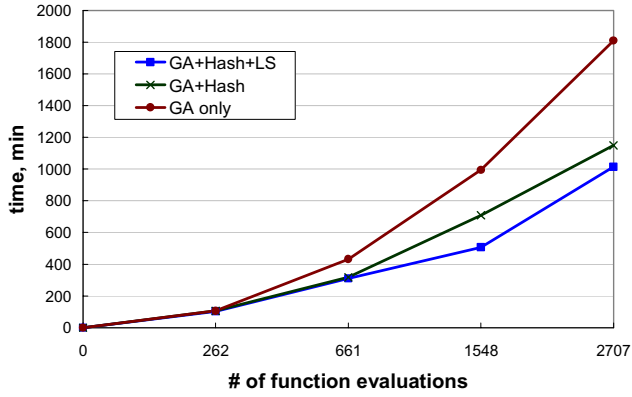

Fig. 7. Optimization speed-up due to candidate solutions caching.

encoding results in two different bit representations of the same physical solution configuration (over-mapping) or when population close to convergence.

In general, if a GA needs to find the fitness (the image quality) of N video designs, out of which only M are not tested before, then we expect to get a gain in computation time:

$$\Delta T = (N - M)\,T_{eval} - N \cdot T_{look-up} - M \cdot T_{update} \tag{10}$$

Here T_{eval} is the time of a single evaluation, $T_{look-up}$ and T_{update} are the times required for finding a record in memory and for writing a record, respectively. The gain is positive if

$$T_{eval} > \frac{N \cdot T_{look-up} + M \cdot T_{update}}{N - M} \tag{11}$$

This is obviously the case when dealing with video systems, where the evaluation time may extend to 6-10 minutes, while the look-up and update time is of the order of nanoseconds. Our implementation of this memory is based on a classical linked list hash. Hash memory has reasonable space requirements for keeping all the visited solutions and their values. Figure 7 shows the actual speed-up of GA variants with solution caching.

7 Conclusion

We presented a method for automatically optimizing a complicated real-world video processing system, without prior information about the constituent video processing components. Using an automatic optimization method necessitates the use of a cost function, which evaluates the perceptual image quality automatically. We introduced a method to combine a number of image quality metrics to maximize the correlation between the perceived quality and the measured objective quality.

We described several approaches to speed up the execution time needed for finding the (sub) optimal solution in a GA driven optimization problem. The solution quality was improved by hybridizing GA with different local search methods, while the run time has been reduced by using memory hash table of previously visited sample points. Several implementations of memetic algorithms have been presented and their performances have been illustrated.

Memetic optimization allows us to keep the versatility provided by GA while speeding up the rate of convergence toward the global optima. In addition, memory-supported GA proves to be very useful, as it avoids re-calculating the objective function especially toward the final convergence.

References

1. Ahumada, A J. Jr.: Computational image quality metrics: A review, in SID Symposium, Digest, vol. 24, (1993) 305-308
2. Baluja, S.: Genetic Algorithms and Explicit Search Statistics, In Mozer, M.C., Jordan, M.I., Petsche, T. (Eds.) Advances in NIPS 9, MIT Press, Cambridge, MA, (1997) 319-325
3. Boschetti, F., Dentith, M., List, R.: Inversion of seismic refraction data using Genetic Algorithms, Geophysics, (1996) 1715-1727
4. Chipperfield, A., Fleming, P.: Parallel Genetic Algorithms, in Parallel and Distributed Computing Handbook, A. Y. H. Zomaya ed., McGraw Hill, (1996) 1118-1143
5. Daly, S.: The visible differences predictor: An algorithm for the assessment of image fidelity, in Digital Images and Human Vision, ed. A. B. Watson, MIT Press, (1993) 179-206
6. Eshelman, L., The CHC Adaptive Search Algorithm. How to Have Safe Search When Engaging in Nontraditional Genetic Recombination, in G. Rawlins, editor, Foundations of Genetic Algorithms, Morgan Kaufmann, (1991) 265-283
7. Goldberg, D. E.: Genetic Algorithms in Search, Optimization and Machine Learning, Reading, Mass. Addison-Wesley, (1989)
8. Goldberg D.E.: Zen and the Art of Genetic Algorithms, In Proc. of the Third Intl Conf. on Genetic Algorithms, Fairfax, VA, ed. by J.D. Schaffer, Morgan Kaufmann, San Mateo CA, 80 (1989)
9. de Haan G., Video Processing for Multimedia Systems, CIP-Data Koninklijke Bibliotheek, The Hague, (2000)
10. Husbands P.: Genetic Algorithms in Optimization and Adaptation, in Advances in Parallel Algorithms, Kronsjo and Shumsheruddin (eds.), (1990) 227-276
11. ITU-R Recommendation 500-7: Methodology for the subjective assessment of the quality of television pictures, ITU, Geneva, Switzerland, (1995)
12. Morris, G.M., Goodsell, D.S., Halliday, R.S., Huey, R., Hart, W.E., Belew, R.K., Olson, A.J.: Automated docking using a Lamarckian genetic algorithm and an empirical binding free energy function. J. Comput. Chem., 19, (1998) 1639-1662
13. Nelder, J. A., Mead, R.: A simplex method for function minimization. Computer Journal, **7** (1965) 308-313
14. Ojo, O.A.: An Algorithm for Integrated noise Reduction and Sharpness Enhancement, Proceedings of International Conference on Consumer Electronics, Los Angeles, (2000) 58-59

15. Quagliarella, D., Vicini, A., Hybrid genetic algorithms as tools for complex optimisation problems, in New Trends in Fuzzy Logic II, Eds. P.Blonda et al., World Scientific, Singapore (1998)
16. Parmee, I.C.: High-Level Decision Support for Engineering Design using the Genetic Algorithms and Complementary Techniques. Developments in Artificial Intelligence for Civil and Structural Engineering, Civil-Comp Press, Edinburgh, UK, (1995), 197-204
17. Pelikan, M., Goldberg, D.E., Lobo, F. : A Survey of Optimization by Building and Using Probabilistic Models, Proceedings of the American Control Conference (ACC-00). Chicago, IL, (2000)
18. Quagliarella, D. and Vicini, A.: Coupling Genetic Algorithms and Gradient Based Optimization Techniques, in Genetic Algorithms and Evolution Strategies in Engineering and Computer Science, Eds. D. Quagliarella et al., Whiley, (1998)
19. Radcliffe, N. J.: Formal memetic algorithms. In Evolutionary Computing: AISB Workshop, T.C. Fogarthy (ed.), Springer Verlag, LNCS 865, (1994) 1-16
20. Renders J. M., Bersini H.: Hybridizing genetic algorithms with hill-climbing methods for global optimization: Two possible ways. In Proceedings of the First IEEE Conference on Evolutionary Computation, (1994) 312-317
21. Scharf L. L.: Statistical Signal Processing: Detection, Estimation, and Time Series Analysis, Addison Wesley Longman, (1991)
22. Ulder, N.L.J., Aarts, E.H.L., Bandelt, H.-J., Van Laarhoven, P.J.M., Pesch, E.: Improving TSP exchange heuristics by population genetics. Proceedings International Workshop on Parallel Problem Solving from Nature, Dortmund, Germany, (1990) 109-116
23. Van den Branden, C. J., Verscheure, O.: Perceptual Quality Measure Using a Spatio-Temporal Model of Human Visual System, Proc. SPIE, vol. 2668, San Jose, CA, (1996) 450-61
24. van Zon K., Ali W.: Automated Video Chain Optimization. IEEE Transactions of ICCE, **47** (3), (2001) 593-603
25. VQEG, Final Report From the Video Quality Experts Group on the Validation of Objective Models of Video Quality Assessment, March 2000. Available at http://www.its.bldrdoc.gov/vqeg/
26. Whitley, D., Gordon, S., Mathias, K.: Larmarckian Evolution, The Baldwin Effect and Function Optimization. In Proc. Parallel Problem Solving from Nature, PPSN III, (1994) 6-15
27. Wolf, S., Pinson, M.: Video Quality Measurement Techniques. National Telecommunications and Information Administration (NTIA) Report 02-392, US Department of Commerce, (2002)
28. Yagiura, M., Ibaraki, T.: On metaheuristic algorithms for combinatorial optimization problems. Systems and Computers in Japan **32**(3), (2001) 33-55
29. Yen, J., Liao, J. C., Randolph, D., Lee, B.: A hybrid approach to modeling metabolic systems using genetic algorithm and simplex method. In Proceedings of the 11th IEEE Conference on Artificial Intelligence for Applications (CAIA95), Los Angeles, CA, (1995) 277-283