COMPARISON OF GENETIC ALGORITHMS AND PARTICLE SWARM OPTIMISATION FOR FERMENTATION FEED PROFILE DETERMINATION

Karl O. Jones

***Abstract**:* In recent years the area of Evolutionary Computation has come into its own. Two of the popular developed approaches are Genetic Algorithms and Particle Swarm Optimisation, both of which are used in optimisation problems. Since the two approaches are supposed to find a solution to a given objective function but employ different strategies and computational effort, it is appropriate to compare their implementation. A study is presented illustrating the performance of both genetic algorithms and particle swarm optimisation, demonstrating their ability to generate a fermentation process feed profile based on a number of objective functions. Results demonstrate how the learning mechanism developed an optimal feed profile which meets the defined criteria.

***Keywords**:* Feed profile, fermentation, genetic algorithm, particle swarm optimisation.

INTRODUCTION

Fermentation processes are associated with the production of yeast, pharmaceuticals, foods and beverages, chemicals, and bulk enzymes. These processes amount to over £1 billion per annum to the UK economy alone, hence there are significant cost incentives for improving the profitability and/or efficiency of the processes by employing modern approaches, such as artificial intelligence techniques. The fermentation of *Saccharomyces cerevisiae* is especially problematical to adequately model and/or control accurately, owing to its intrinsic time varying and non-linear dynamics. The process should be controlled such that the maximum biomass is produced in the shortest time using the minimum raw materials, such as substrate and oxygen. The cost of the various components of a growth medium can have a significant effect on the overall cost of fermentation processes since they can account for between 38% and 73% of total production costs. The organic carbon source is often the most expensive component. Ratledge [1] has made a detailed analysis of annual price and availability of major carbon substrates. In fed-batch fermentations, the embedded exponential growth pattern requires a corresponding substrate supply. This substrate demand can pose a considerable problem for a non-optimised substrate feed profile. Any excess substrate is a waste of resources, while substrate lack is growth limiting factor. In theory the optimal feed profile is an exponential increase matching the demand from the increasing cell numbers. This type of profile can be difficult to achieve since many industrial fermentation processes do not have sophisticated feed pumps: instead they use pumps which deliver at a fixed rate for a set length of time (that is, quantized levels). It is possible to adapt the feed profile so that a high biomass yield is obtained whilst minimising the total substrate supplied. One approach to formulating a more optimised feed profile is to utilise Artificial Intelligence techniques such as Particle Swarm Optimisation (PSO) or Genetic Algorithms (GA).

In the 1950s computer scientists studied evolutionary systems as optimisation tools, introducing the basics of evolutionary computing. Until the 1960s, the field of evolutionary systems was working in parallel with GA research. When they started to interact, a new field of evolutionary programming appeared by introducing new concepts of evolution, selection and mutation. Holland [2] defined the concept of the GA as a metaphor of the Darwinian theory of evolution applied to biology. Implementation of a GA begins with a population of random chromosomes. The algorithm then evaluates these structures and allocates reproductive opportunities such that chromosomes which represent a better solution to the problem are given more chance to "reproduce". In selecting the best candidates, new fitter offspring are produced and reinserted, and the less fit removed. Like

neural networks, GAs are based on a biological metaphor, however, instead of the biological brain, GAs view learning in terms of competition among a population of evolving, alternative concepts. A GA maintains a population of candidate problem solutions. Based on their performance, the fittest of these solutions not only survive, but, through an analogy with sexual reproduction, exchange information with other candidates to form new solutions. In using operators such as crossover and mutation the chromosomes exchange their characteristics. The suitability of a solution is typically defined with respect to the current population [3]. GA techniques have a solid theoretical foundation [3], based on the Schema Theorem [2].

The implicit rules followed by the members of fish schools and bird flocks, that allow them to undertake synchronized movement, without colliding, has been studied by several scientists [4]. There is a general belief that social sharing of information among individuals of a population, may provide an evolutionary advantage, and there are numerous examples coming from nature to support this. This was the core idea behind the development of PSO. The PSO method is a member of the wide category of Swarm Intelligence methods [5]. Kennedy originally proposed PSO as a simulation of social behaviour, and it was initially introduced as an optimisation method in 1995 [6]. PSO can be easily implemented and is computationally inexpensive since its memory and CPU speed requirements are low [7]. Furthermore, it does not require gradient information of the objective function being considered, only its values. PSO has proved to be an efficient method for numerous general optimisation problems, and in some cases it does not suffer from the problems encountered by other Evolutionary Computation techniques [6]. PSO has been successfully applied to a range of problems, from function optimisation to the training of neural networks. Although, while PSO typically moves quickly towards the best general area in the solution space for a problem, it often has difficulty in making the fine grain search required to find the absolute best point.

GENETIC ALGORITHM OPERATION

To illustrate the working process of genetic algorithm, the steps to realise a basic GA are listed:

Step 1: Represent the problem variable domain as a chromosome of fixed length; choose the size of the chromosome population N, the crossover probability P_c and the mutation probability P_m.

Step 2: Define a fitness function to measure the performance of an individual chromosome in the problem domain. The fitness function establishes the basis for selecting chromosomes that will be mated during reproduction.

Step 3: Randomly generate an initial population of size N: $x_1, x_2, ..., x_N$

Step 4: Calculate the fitness of each individual chromosome: $f(x_1), f(x_2), ..., f(x_N)$

Step 5: Select a pair of chromosomes for mating from the current population. Parent chromosomes are selected with a probability related to their fitness. High fit chromosomes have a higher probability of being selected for mating than less fit chromosomes.

Step 6: Create a pair of offspring chromosomes by applying the genetic operators.

Step 7: Place the created offspring chromosomes in the new population.

Step 8: Repeat Step 5 until the new population size equals that of the initial population, N.

Step 9: Replace the initial (parent) chromosome population with the new (offspring) population.

Step 10: Go to Step 4, and repeat the process until the termination criterion is satisfied.

A GA is an iterative process. Each iteration is called a generation. A typical number of generations for a simple GA can range from 50 to over 500. Common practice is to

terminate a GA after a specified number of generations and then examine the best chromosomes. If no satisfactory solution is found, then the GA is restarted.

PARTICLE SWARM OPTIMISATION ALGORITHM OPERATION

PSO optimises an objective function by undertaking a population-based search. The population consists of potential solutions, named particles, which are a metaphor of birds in flocks. These particles are randomly initialised and freely fly across the multi-dimensional search space. During flight, each particle updates its own velocity and position based on the best experience of its own and the entire population. The updating policy drives the particle swarm to move toward the region with the higher objective function value, and eventually all particles will gather around the point with the highest objective value. The detailed operation of particle swarm optimisation is given below:

Step 1: Initialisation. The velocity and position of all particles are randomly set to within pre-defined ranges.

Step 2: Velocity Updating. At each iteration, the velocities of all particles are updated according to:

$$\vec{v}_i = w\vec{v}_i + c_1 R_1(\vec{p}_{i,best} - \vec{p}_i) + c_2 R_2(\vec{g}_{i,best} - \vec{p}_i) \qquad (1)$$

where $\vec{p}_i$ and $\vec{v}_i$ are the position and velocity of particle *i*, respectively; $\vec{p}_{i,\text{best}}$ and $\vec{g}_{i,best}$ is the position with the 'best' objective value found so far by particle *i* and the entire population respectively; *w* is a parameter controlling the flying dynamics; R_1 and R_2 are random variables in the range [0, 1]; c_1 and c_2 are factors controlling the related weighting of corresponding terms. The inclusion of random variables endows the PSO with the ability of stochastic searching. The weighting factors, c_1 and c_2, compromise the inevitable trade-off between exploration and exploitation. After updating, $\vec{v}_i$ should be checked and secured within a pre-specified range to avoid violent random walking.

Step 3: Position Updating. Assuming a unit time interval between successive iterations, the positions of all particles are updated according to:

$$\vec{p}_1 = \vec{p}_i + \vec{v}_i \qquad (2)$$

After updating, $\vec{p}_i$ should be checked and limited to the allowed range.

Step 4: Memory updating. Update $\vec{p}_{i,\text{best}}$ and $\vec{g}_{i,best}$ when condition is met.

$$\begin{aligned} \vec{p}_{i,best} &= \vec{p}_i \qquad \text{if } f(\vec{p}_i) > f(\vec{p}_{i,best}) \\ \vec{g}_{i,best} &= \vec{g}_i \qquad \text{if } f(\vec{g}_i) > f(\vec{g}_{i,best}) \end{aligned} \qquad (3)$$

where $f(\vec{x})$ is the objective function subject to maximization.

Step 5: Termination Checking. The algorithm repeats Steps 2 to 4 until certain termination conditions are met, such as a pre-defined number of iterations or a failure to make progress for a certain number of iterations. Once terminated, the algorithm reports the values of $\vec{g}_{best}$ and $f(\vec{g}_{best})$ as its solution.

APPLICATION AND RESULTS

Genetic Algorithm

Extensive simulation tests have been conducted on the GA and PSO to test the effectiveness of the mechanisms, using simulations of a model of a fed-batch fermentation of *S. cerevisiae* (Bakers' Yeast), where the model is based on stoichiometric constants and kinetic equations [8]. For all the tests, the feed rate was defined to be in the range 0-200 mg/h, with each time period lasting 15 minutes. Table 1 illustrates the objective functions utilised for the simulation tests. The tests performed held most elements of the GA constant while one element was changed: for example, four different minimisation functions were used while maintaining a population of 200 individuals with a crossover rate of 0.8 and a mutation rate of 0.0225.

Table 1 Objective Functions Utilised

1. $f(X_{Actual}, X_{Theory})$
2. $f(T_S, X_{Actual})$
3. $f(S)$
4. $f(S, X_{Actual}, X_{Theory})$
5. $f(T_S, X_{Actual}, X_{Theory})$

Since GAs are stochastic, their performance usually varies from generation to generation. The first objective function considers the difference between the actual cell concentration (X_A) and theoretical maximum cell concentration (X_T). The developed feed profile is somewhat high for the whole fermentation period (Figure 1), with a consequential excess substrate in the broth. The cell concentration has an ideal increase for the complete fermentation period, achieving the theoretical maximum value. The second objective function considers the ratio of the Total Substrate Feed (T_S) and X_A at the end of the fermentation period. In this instance, the feed profile has a lower rate than the first test. While there are periods of excess substrate in the broth, basically the substrate is kept to a minimum, however the final cell concentration is much reduced versus the theoretical maximum (Figure 2). The third objective function considers only the Substrate concentration over the fermentation period. The general level of the feed profile is higher than that for test 2 and does exhibit a general increase over time. The cell concentration increases over the fermentation period, although its final value is only half that the theoretical maximum (Figure 3). The final objective function considers the ratio of the substrate concentration and the difference between X_A and X_T. The feed profile is lower than that for tests 1 and 3. Besides a couple of points when there is excessive substrate, generally the substrate is at a minimal. The final cell concentration is less than that for test 3 (Figure 4).

Particle Swarm Optimisation

The tests performed held the elements of the PSO constant while the minimisation function was altered: the parameters used were a swarm of 200 individuals, using two neighbour observations, and a velocity weight of 0.95 at the start of PSO iterations reducing to 0.4 for the final iteration. The first objective function (difference between X_A and X_T). The profile developed is fairly high throughout the fermentation (Figure 5), resulting in a significant amount of excess substrate in the broth. The cell concentration has a perfect increase for the complete fermentation period. The final cell concentration achieves the maximum theoretical value. The second objective function (using T_S and final X_A value). In this case, the developed feed profile has a generally low rate with significant time with an almost zero feed rate. This results in a very low level of excess substrate in the broth. The cell concentration throughout the fermentation is significantly below the theoretical maximum (Figure 6) achieving only a 50% level. Overall, this is a poor profile for the fermentation although the PSO has followed the objective function requirements. The third objective function (considering S over the fermentation). The developed feed profile has a steady increase; with a corresponding increase in cell concentration. The final cell concentration is high although not quite as high as the theoretical maximum (Figure 7). In this case the PSO has performed an admirable task. The fourth objective function (ratio of S and difference between X_A and X_T). The feed profile developed exhibits a general increase throughout the fermentation (Figure 8). There is a resultant minimal amount of excess substrate in the broth, and a steady increase in cells although below the theoretical, finally reaching around 70% of the maximum.

Comparison

For each methodology applied to each test, it is clear that the required objective function has been achieved. However the results seem to indicate that the feed profiles formed by the Genetic Algorithm approach are superior to those produced by the Particle Swarm Optimisation: that is that generally the final cell concentration is higher and the excess substrate level is lower which is the fundamental requirement of the fermentation system.

DISCUSSION

The strength of GAs is in the parallel nature of their search. A GA implements a powerful form of hill climbing that preserves multiple solutions, eradicates unpromising solutions, and provides reasonable solutions. Through genetic operators, even weak solutions may continue to be part of the makeup of future candidate solutions. The genetic operators used are central to the success of the search. All GAs require some form of recombination, as this allows the creation of new solutions that have, by virtue of their parent's success, a higher probability of exhibiting a good performance. In practice, crossover is the principal genetic operator, whereas mutation is used much less frequently. Crossover attempts to preserve the beneficial aspects of candidate solutions and to eliminate undesirable components, while the random nature of mutation is probably more likely to degrade a strong candidate solution than to improve it. Another source of the algorithm's power is the implicit parallelism inherent in the evolutionary metaphor. By restricting the reproduction of weak candidates, GAs eliminate not only that solution but also all of its descendants. This tends to make the algorithm likely to converge towards high quality solutions within a few generations.

Particle Swarm Optimisation shares many similarities with evolutionary computation (EC) techniques in general and GAs in particular. All three techniques begin with a group of a randomly generated population, all utilise a fitness value to evaluate the population. They all update the population and search for the optimum with random techniques. A large inertia weight facilitates global exploration (search in new areas), while a small one tends to assist local exploration. The main difference between the PSO approach compared to EC and GA, is that PSO does not have genetic operators such as crossover and mutation. Particles update themselves with the internal velocity, they also have a memory that is important to the algorithm. Compared with EC algorithms (such as evolutionary programming, evolutionary strategy and genetic programming), the information sharing mechanism in PSO is significantly different. In EC approaches, chromosomes share information with each other, thus the whole population moves like one group towards an optimal area. In PSO, only the 'best' particle gives out the information to others. It is a one-way information sharing mechanism, the evolution only looks for the best solution. Compared with ECs, all the particles tend to converge to the best solution quickly even in the local version in most cases. Compared to GAs, the advantages of PSO are that PSO is easy to implement and there are few parameters to adjust.

CONCLUSIONS

Techniques such as PSO and Genetic Algorithms are inspired by nature, and have proved themselves to be effective solutions to optimization problems. However, these techniques are not a panacea, despite their apparent robustness. There are control parameters involved in these meta-heuristics, and appropriate setting of these parameters is a key point for success. In general, some form of trial-and-error tuning is necessary for each particular instance of optimization problem. Additionally, any meta-heuristic should not be thought of in isolation: the possibility of utilising hybrid approaches should be

considered. Additionally for both approaches the major issue in implementation lies in the selection of an appropriate objective function.

Like many AI approaches, both PSO and GA are computationally intensive, and the tuning process tends to be *ad hoc*. The next stage of research would involve implementing the methodology on a real-time process, although the work conducted so far suggests that the procedure will operate successfully. In conclusion, the work presented illustrates that Genetic Algorithms and Particle Swarm Optimization can be used for generating feed profiles that are optimised for a given objective function. The main problem in implementation lies in the selection of an appropriate objective function, then once the control parameters have been tuned both GA and PSO can produce a result.

REFERENCES

[1] Ratledge, C. 1977. Fermentation substrates. *Ann. Rep. Ferm. Processes*. **1**, 49-71.
[2] Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems*. University of Michigan Press.
[3] Whitley, D. (1993). A Genetic Algorithm Tutorial. Computer Science Department, Colorado State University Fort Collins, CO 80523, USA.
[4] Heppener, F. and U. Grenander. 1990. A stochastic nonlinear model for coordinate bird flocks. In Krasner, S. (Ed.) *The Ubiquity of Chaos*. AAAS Publications, Washington DC.
[5] Kennedy, J. and R.C. Eberhart. 2001. Swarm Intelligence. Morgan Kaufman Publishers.
[6] Kennedy, J. and R.C. Eberhart. 1995. Particle Swarm Optimisation. Proceedings IEEE International Conference on Neural Networks, IV, p. 1942-1948.
[7] Eberhart, R.C., Simpson, P. and Dobbins, R. 1996. Computational Intelligence PC Tools. Academic Press.
[8] Pirt, S.J. 1975. *Principles of microbe and cell cultivation*. Blackwell Scientific Publications, London.

FIGURES

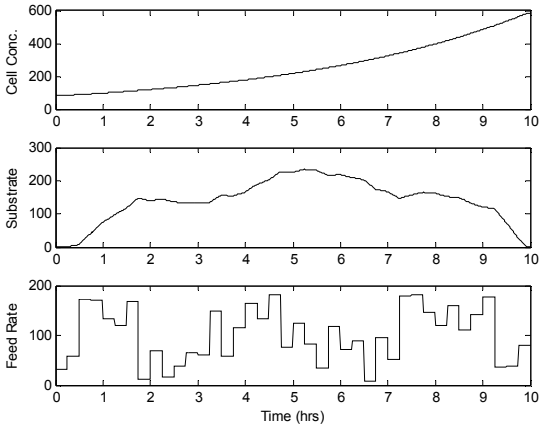

Figure 1 Results for objective function 1 (GA).

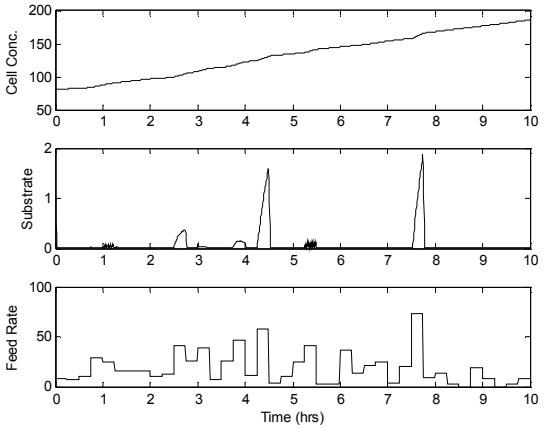

Figure 2 Results for objective function 2 (GA).

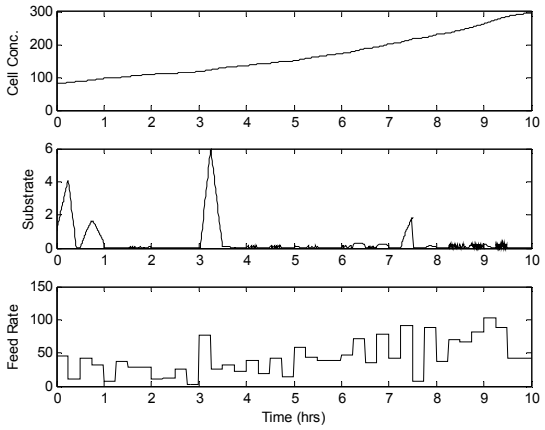

Figure 3 Results for objective function 3 (GA).

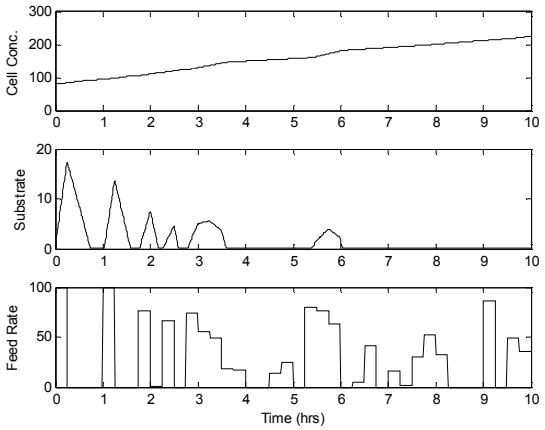

Figure 6 Results for objective function 2 (PSO).

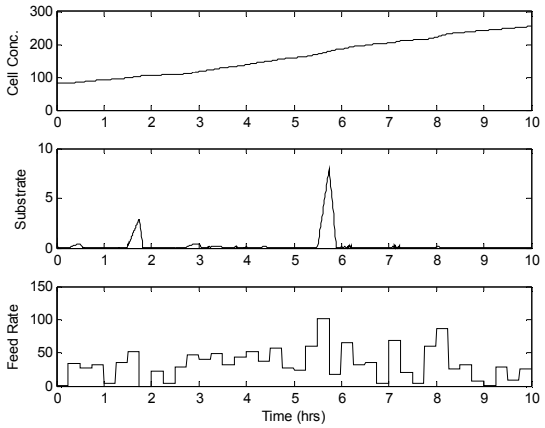

Figure 4 Results for objective function 4 (GA).

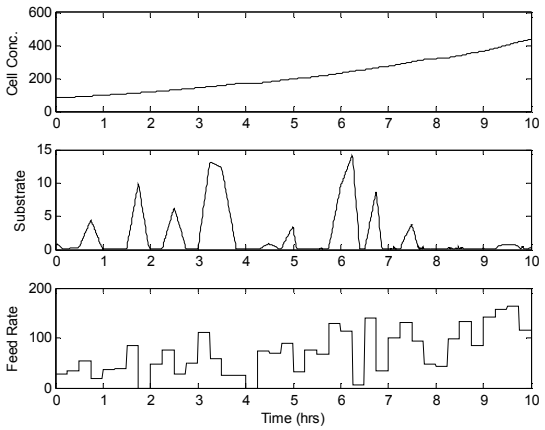

Figure 7 Results for objective function 3 (PSO).

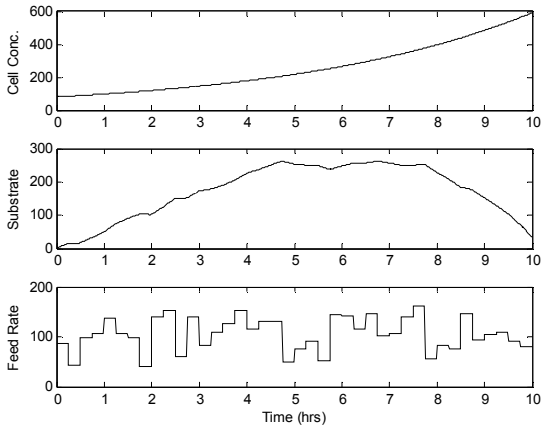

Figure 5 Results for objective function 1 (PSO).

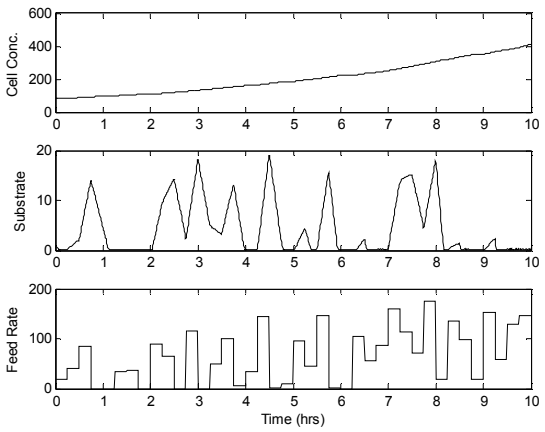

Figure 8 Results for objective function 4 (PSO).

ABOUT THE AUTHOR

Dr. Karl O. Jones is from the School of Engineering, Liverpool John Moores University, Liverpool, United Kingdom. His interests are in the application of artificial intelligence techniques to control and biotechnology problems. Phone: +44 151 231 2199, E-mail: k.o.jones@livjm.ac.uk