# A Fast and Elitist Multi-objective Particle Swarm Algorithm: NSPSO

*Yang Liu*
*Manchester Interdisciplinary Biocentre,*
*Faculty of Engineering and Physical Sciences,*
*University of Manchester, Manchester, UK*
*Yang.Liu@manchester.ac.uk*

## Abstract

*In this paper, a new nondominated sorting particle swarm optimisation (NSPSO), is proposed, that combines the operations (fast ranking of non-dominated solutions, crowding distance ranking and elitist strategy of combining parent population and offspring population together) of a known MOGA NSGA-II and the other advanced operations (selection and mutation operations) with a single particle swarm optimiser (PSO). The efficacy of this algorithm is demonstrated on 2 test functions, and the comparison is made with the NSGA-II and a Multi-objective PSO (MOPSO-CD). The simulation results suggest that the proposed optimisation framework is able to achieve good solutions as well diversity compared to NSGA-II and MOPSO-CD optimisation framework.*

## 1. Introduction

Many real-world optimisation problems require the process of simultaneous optimisation of possibly conflicting multiple objectives, and this is termed multi-objective optimisation. The multi-objective biological model calibration problem can be stated as the optimisation problem:

$$\text{Minimize } F(\theta)=\{f_1(\theta),f_2(\theta),...,f_m(\theta)\} \quad (1)$$

Where $f_1(\theta), f_2(\theta), ..., f_m(\theta)$ are the $m$ non-commensurable objective functions to be simultaneously minimized with respect to the parameters $\theta$ of the model [1]. By definition, the multi-objective optimisation has a very different nature from that of single-objective optimisation. Unlike single-objective optimisation where only one optimal solution is pursued, a typical multi-objective optimisation problem produces a set of solutions which are superior to the rest of the solutions with respect to all objective criteria but are inferior to other solutions in one or more objectives. These solutions are known as Pareto optimal solutions or non-dominated solutions. In absence of additional information, it is not possible to distinguish any one of the Pareto solutions as being objectively better than any others with respect to all the objectives concerned (i.e. there is no uniquely "best" solution); therefore, any one of them is an acceptable solution [1]. Once the set of optimal solutions is identified, the designers have the freedom of choosing one solution out of many possible solutions based on their experience and prior knowledge and other criteria or constraints.

Kennedy and Eberhart developed particle swarm optimisation based on the analogy of swarming animals, such as a flock of birds or school of fish [2]. In each iteration, each agent is updated with reference to two "best" values: *pbest* is the best solution (in terms of fitness) the individual particle has achieved so far, while *gbest* is the best obtained globally so far by any particle in the population. Each agent seeks to modify its position using the current positions, the current velocities, the distance between the current position and *pbest*, and the distance between the current position and *gbest.* Compared to genetic algorithm optimisation, there are not many parameters that need to be tuned in PSO. The parameters are: the number of particles; weighting factors; and the maximum change for a particle. It is generally found that operation is not very sensitive to parameter settings. For the number of particles, the typical range is 20 – 40 [3]. The weighting factors, $c_1$ and $c_2$, are often to 2, though other settings are used in different papers, typically with $c_1 = c_2$ and in the range [0, 3].

PSO seems particularly suitable for multi-objective optimisation mainly because of the high speed of convergence that that algorithm presents for single objective optimisation [4]. Among those algorithms that extend PSO to solve multi-objective optimisation problems [4, 5, 6, 7] they needed one external archive to save non-dominated solutions as *gbests* and the second one to save *pbests*. This paper proposed a novel hybrid approach through crossing over the PSO and advanced operations, called elitist Non-dominated Sorting Particle Swarm Optimisation (NSPSO). We proposed a novel selection regime for the choosing of global best (*gbest*) and personal best (*pbest*) for swarm members in multi-objective particle swarm optimisation (MOPSO) without using external

archives. It means the algorithm is simple and computer coding is easy to implement to any optimisation problems. The NSPSO combines the advanced operations (fast ranking of non-dominated solutions, crowding distance ranking, elitist strategy of combining parent population and offspring population together, selection and mutation operations) with a single particle swarm optimisation (PSO). In the paper, we investigate the performance of the NSPSO by adding these operators. Simulations for the test functions show that the proposed NSPSO method possesses better ability to finding the optimal Pareto front compared to the NSGA-II and MOPSO-CD.

## 2. Elitist non-dominated sorting genetic algorithm (NSGA-II)

The capabilities of multi-objective genetic algorithms (MOGAs) to explore and discover Pareto-optimal fronts on multi-objective optimisation problems have been well recognized. It has been shown that MOGAs outperform traditional deterministic methods to this type of problem due to their capacity to explore and combine various solutions to find the Pareto front in a single run. We will implement a multi-objective optimisation technique called the Non-Dominated Sorting Genetic Algorithm II (NSGA-II), which is described in detail by Deb et al. [8]. The NSGA-II algorithm may be stated as follows:

**(1)** Create a random parent population of size N;
**(2)** Sort the population based on the nondomination;
**(3)** Assign each solution a fitness (or rank) equal to its nondomination level (minimisation of fitness is assumed);
**(4)** Use the usual binary tournament selection, recombination, and mutation operators to create a new offspring population of size N;
**(5)** Combine the offspring and parent population to form extended population of size 2N;
**(6)** Sort the extended population based on nodomination;
**(7)** Fill new population of size N with the individuals from the sorting fronts starting from the best;
**(8)** Invoke the crowding comparison operator to ensure diversity if a front can only partially fill the next generation (This strategy is called "niching");
**(9)** Repeat the steps (2) to (8) until the stopping criterion is met. The stopping criterion may be a specified number of generations.

It is clear from the above description that NSGA-II uses (i) a fast non-dominated sorting approach, (ii) an elitist strategy, and (iii) no niching parameter [8].

## 3. Elitist non-dominated sorting particle swarm optimisation (NSPSO)

The goal of our multi-objective hybrid algorithm is to combine single-objective PSO with NSGA-II operations without loosing performance on establishing the Pareto-front. The NSPSO combines the strengths of the these advanced operations (A fast non-dominated sorting approach, crowding distance ranking, elitist strategy, mutation and selection operations) with single-objective PSO search. The hybrid algorithm is presented below:

**Step 1:** Generate an initial population $P$ (Population size $= N$) and velocity for each individual (agent or particle) in a feasible space; Set the maximum speed $v_i^{\max}$ ($v_i^{\max}$ = its upper bound minus lower bound) for a variable.
**Step 2:** Sort the population based on the non-domination and crowding distance ranking.
**Step 3:** Do rank-based selection operator [9].
**Step 4:** Assign each individual a fitness (or rank) equal to its non-domination level (minimisation of fitness is assumed).
**Step 5:** Randomly choose one individual as *gbest* for N times from the nondominated solutions, and modify each searching point using previous PSO formula and the *gbest*:

$$v_i^{k+1} = K[v_i^k + c_1 \times rand() \times (pbest_i - s_i^k) + c_2 \times rand() \times (gbest - s_i^k)] \quad (2)$$

$$K = \frac{2}{\left|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}\right|} \quad \text{where } \varphi = c_1 + c_2 \ \ \varphi > 4 \quad (3)$$

$$s_i^{k+1} = s_i^k + v_i^{k+1} \quad (4)$$

where *rand()* is a random number between (0, 1). The constriction factor approach can generate higher quality solutions than the conventional PSO approach [13]. If current position outside the boundaries, then it takes the upper bound or lower bound and its velocity is generated randomly ($0 \le v_i^{k+1} \le v_i^{\max}$) and multiplied by -1 so that it searches in the opposite direction.
**Step 6:** Do mutation operator [4, 10].
**Step 7:** Combine the offspring and parent population to form extended population of size 2N.
**Step 8:** Sort the extended population based on non-domination and fill the new population of size N with individuals from the sorting fronts starting to the best.
**Step 9:** Modify the $pbest_i$ of each searching point:

If current rank of the new individual (offspring) $P_i^{k+1}$ is smaller than or equal to the previous one (parent) in $R$, replace the $pbest_i$ with current individual; otherwise keep the previous $pbest_i$.

**Step 10:** Perform step (2) to (9) until the stopping criterion is met.

The main differences of our approach with respect to the other proposals existing in the literature [4, 5, 6] are:

- We add selection operator to the multi-objective particle swarm algorithm.
- We don't need external repository to save the *pbest* and *gbest*.
- Selection regime for the choosing of global best (*gbest*) and personal best (*pbest*) for swarm members is based on elitist operation.
- The programming code is much short compared with the other proposals and simple to implement to any optimisation problem.

## 4. Optimisation framework

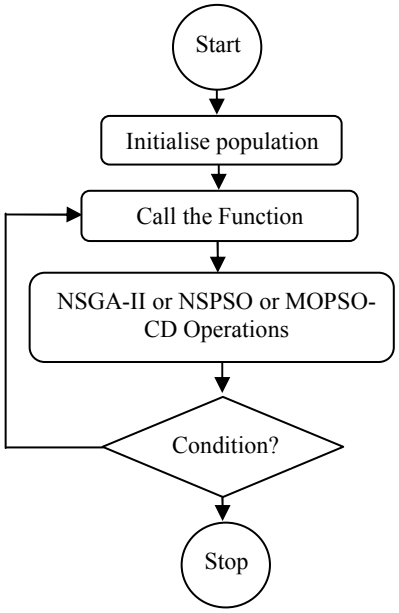



**Figure 1:** Outline of optimisation process.

The general flow chart for the optimisation process using NSGA-II, MOPSO-CD and NSPSO for the function is presented in Figure 1. The calibration process can be performed via an automatic process. In order to do so, the user may need to write two small programs for the process. The fist program is used to change the parameters of the objective function and the second program is used to calculate objective function. As the standard search progresses, the entire population tends to converge to the global Pareto front. This process is continued until a satisfied condition is met. The termination criterion for the iterations is determined according to whether the max iteration or a designed value of the fitness is reached.

## 5. Performance metrics

A number of performance metrics have been suggested in the past [8, 11]. Two goals are usually considered for a multi-objective optimisation. First, the population should converge towards the optimal Pareto front. Second, the optimal Pareto front should have maximum spread of solutions. Based on this notion, we adopted S metric to evaluate each of two aspects. A definition of the S metric is given in [11]. The S metric calculates the hypervolume of the multi-dimensional region enclosed by A and a 'reference point', hence computing the size of region A dominates. It is independent (although needs a reference point to be chosen), so it induces a complete ordering, and it is non-cardinal. Figure 2 illustrates the Procedure of S metric calculation for an optimal Pareto front where two objective ( $f_1, f_2$ ) are to be minimized.

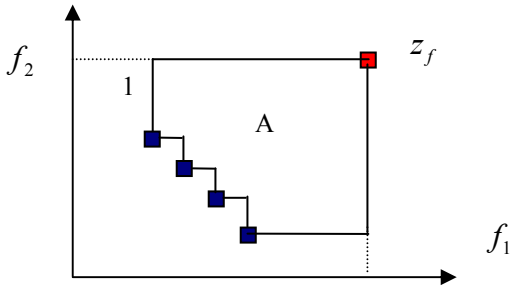



**Figure 2:** The relative value of the S metric upon an arbitrary choice of reference point.

## 6. Application example

In the following examples, the total number of fitness function evaluations was set to 5000. The relevant experiment parameters using the NSPSO for the two test functions are listed in Table 1. For each test, a number of generations equal to 50 were employed as a stopping criterion for NSPSO, NSGA-II and MOPSO-CD when a population of p=100 was used. We used distribution indices for crossover and mutation operations as $\eta_c$=20 and $\eta_m$ = 20 for the real coded NSGA-II. In our research we used the NSPSO developed in Matlab. The source code of NSGA-II (Matlab code) and MOPSO (C code) are

available from [12, 13]. In all the following examples, we report the results obtained from performing 10 random runs of each algorithm compared.

**Table 1:** Experimental parameters

| Parameter | Description | Range |
|---|---|---|
| $c_1$ | Weighting factor 1 | 2.3 |
| $c_2$ | Weighting factor 2 | 2.3 |
| $G$ | The total iterations | 50 |
| $M$ | Mutation Rate | 0.5 |
| $P$ | The number of particles | 100 |

## 6.1 Test function 1

Our first test function was proposed by Schaffer's study [14]
Minimize

$$f_1 = x^2 \qquad (5)$$

Minimize

$$f_2 = (x-2)^2 \qquad (6)$$

Where

$$-10000 \le x \le 10000$$

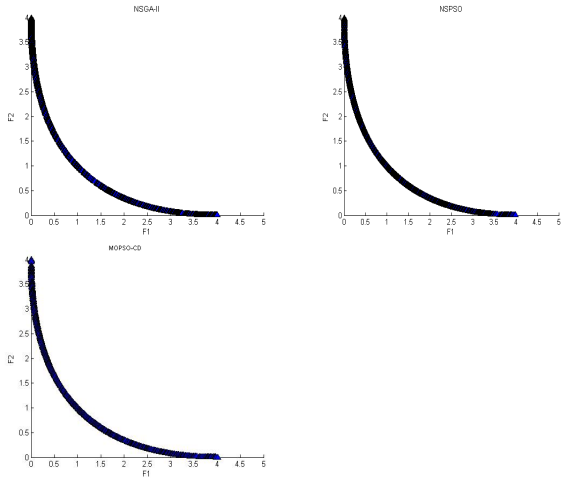



**Figure 3:** Pareto fronts produced by the NSGA-II, NSPSO for the first function

Figure 3 shows the graphical results produced by NSGA-II, NSPSO, and MOPSO-CD. Table 2 shows the comparison of results among the three algorithms considering the metric previously described. The 'O' denotes the algorithm failed to find acceptable optimal Pareto front based on reference point for a random run. It can be seen that the average performance of NSPSO is the best with respect to the S metric. The NSGA-II has the worst result and stability for 10 random runs.

**Table 2:** Results of the S metric for the first function using the three optimisation algorithms

| Trail ( S metric) Reference point (5, 4) | NSGA-II | MOPSO-CD | NSPSO |
|---|---|---|---|
| 1 | **O** | 17.3214 | 17.3287 |
| 2 | **O** | 17.3214 | 17.3283 |
| 3 | **O** | 17.3214 | 17.3283 |
| 4 | **O** | 17.3215 | 17.3282 |
| 5 | 16.9287 | 17.3216 | 17.3287 |
| 6 | **O** | 17.3216 | 17.3285 |
| 7 | 7.4537 | 17.3215 | 17.3278 |
| 8 | 17.3246 | 17.3214 | 17.3283 |
| 9 | 17.3254 | 17.3216 | 17.3281 |
| 10 | **O** | 17.3214 | 17.3274 |
| Mean | | 17.3215 | 17.3282 |
| STD | | 9.18e-005 | 3.9735e-004 |

## 6.2 Test function 2

Our second test function was proposed by Fonseca and Fleming's study [15]
Minimize

$$f_1 = 1 - \exp\left(-\sum_{i=1}^{3}\left(x_i - \frac{1}{\sqrt{3}}\right)^2\right) \qquad (7)$$

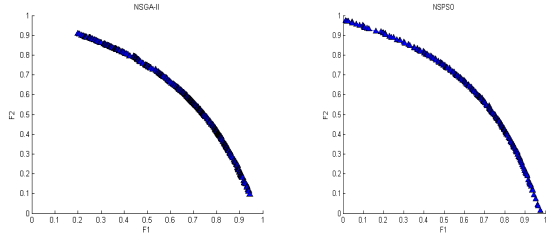
Minimize

$$f_2 = 1 - \exp\left(-\sum_{i=1}^{3}\left(x_i + \frac{1}{\sqrt{3}}\right)^2\right) \qquad (8)$$

Where

$$-4 \le x_1, x_2, x_3 \le 4$$

Figure 4 shows the graphical results produced by NSGA-II, NSPSO, and MOPSO-CD. Table 3 shows the comparison of results among the three algorithms considering the metric previously described. It can be seen that the average performance of MOPSO-CD is the best with respect to the S metric, and the MOPSO-CD has good stability for 10 random runs. However, its result is close to our proposed NSPSO. The NSGA-II has the worst result and stability for 10 random runs.



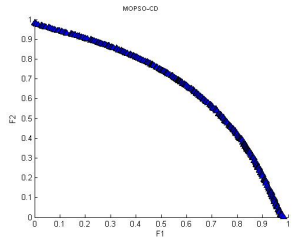



**Figure 4:** Pareto fronts produced by the NSGA-II, NSPSO and MOPSO-CD for the second function.

**Table 3:** Results of the S metric for the second function using the three optimisation algorithms

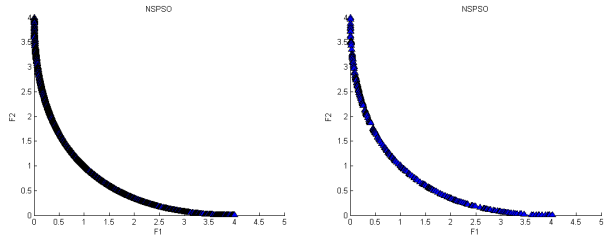
| Trail ( S metric) Reference point (1, 1) | NSGA-II | MOPSO-CD | NSPSO |
|---|---|---|---|
| 1 | 0.2920 | 0.3385 | 0.3343 |
| 2 | 0.2338 | 0.3388 | 0.3356 |
| 3 | 0.2801 | 0.3386 | 0.3347 |
| 4 | 0.3232 | 0.3385 | 0.3345 |
| 5 | 0.3033 | 0.3389 | 0.3349 |
| 6 | 0.2722 | 0.3385 | 0.3357 |
| 7 | 0.3200 | 0.3386 | 0.3344 |
| 8 | 0.3033 | 0.3386 | 0.3348 |
| 9 | 0.3091 | 0.3387 | 0.3340 |
| 10 | 0.2813 | 0.3384 | 0.3354 |
| Mean | 0.2918 | 0.3386 | 0.3348 |
| STD | 0.0265 | 1.5239e-004 | 5.7359e-004 |

## 7. NSPSO operation analysis

In order to analyze effect of the following operators in the NSPSO algorithm, we take the SCH as test function and compare the results running from the same random seed. We use a population size of 100 for a maximum of 50 iterations.

### 7.1 Selection Operator

Figure 5 demonstrates the ability of adding the rank-based selection operation to the NSPSO can converge towards to a better optimal Pareto front.



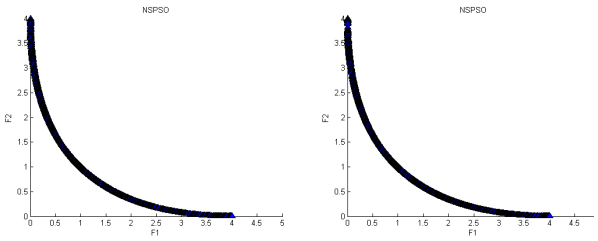

(a) S metric value = 17.3279 with selection operation

**(b)** S metric value = 17.3019 without selection operation

**Figure 5:** Pareto fronts produced by the NSPSO for the test function with and without selection operator.

### 7.2 Updating operator

Figure 6 shows the results of a random run using NSPSO with and without external archive. With regard to the coverage metric, the results show that our proposed NSPSO without losing much accuracy compared to the NSPSO using external archive to save all non-dominated solutions during optimisation process. This also can be seen from the above comparisons using the two different functions.

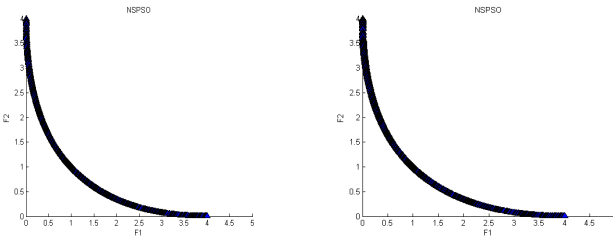



(a) S metric value = 17.3279 without external archive.

(b) S metric value = 17.3286 using external archive.

**Figure 6:** Pareto fronts produced by the NSPSO for the test function with and without external archive.

Figure 7 shows that the mutation operation finds a slighter better set of non-dominated solutions in SCH. Here, we set the mutation rate is equal to 0.5.




(a) S metric value = 17.3279 with mutation operation

(b)S metric value =17.3275 without mutation operation

**Figure 7:** Pareto fronts produced by the NSPSO for the test function with and without mutation operator.

## 8. Conclusion

In this paper, we have proposed a computational fast and elitist multi-objective particle swarm optimisation based on non-dominated sorting approach. In particular, the present methodologies integrate NSGA-II operations and the other advanced operations with the single-objective PSO optimiser. It seems that there are few reported multi-objective PSO that can effectively handle a multi-objective problem.

It is not because it's difficult to choose the *gbest* and *pbest* to construct the multi-objective particle, but also the diversity among the Pareto front and elitist strategy. In this study, we suggested the NSPSO algorithm that can successfully treat multiple objectives optimisation problem. The method was based on the notion of single-objective PSO and on operations drawn from NSGA-II optimisation framework and the other optimisation frameworks. We proposed a novel selection regime for the choosing of global best (*gbest*) and personal best (*pbest*) for swarm members in multi-objective particle swarm optimisation (MOPSO) without using external archives. The hybrid algorithm to guarantee a sufficiently accurate Pareto set in most cases using the NSPSO has been reported. On two test functions, it has been found that the proposed NSPSO has been able to maintain a closer spread of solutions and convergence in the obtained non-dominated front compared to NSGA-II and MOPSO-CD. The worst performance is observed with NSGA-II using 5000 evaluations. Our proposed NSPSO performed close results as MOPSO-CD. The experiments showed that by using NSPSO, it achieved a sufficiently accurate Pareto set and a good diversity in the obtained front compared to NSGA-II and MOPSO-CD.

It is still an open question which weighting factors should be chosen for multi-objective optimisation. It is important to choose the proper values to update agents, so that a small number of evaluations will be yielded. Here, we used the constriction factor to update the velocity. A method to adjust weighting factors of agents without producing additional model parameters or less sensitive parameter and to add advanced operations needs to be investigated in the future.

## 8. References


[1] Yapo P.O., Gupta H.V., Sorooshian S., "Multi-objective global optimisation for hydrologic models", Journal of Hydrology, 1998, 204, pp. 83-97.

[2] Kennedy J., and Eberhart R., "Particle Swarm Optimisation", in Proc. of the IEEE Int. Conf. on Neural Networks, 1995, pp.1942–1945.

[3] Hu X.H., "Particle Swarm Optimisation tutorial", www.swarmintelligence.org/tutorials.php, available online on 26th, Jun. 2007.

[4] Coello C.A.C, "Handling multiple objectives with particle swarm optimization", IEEE Transactions on Evolutionary Computation, 2004, 8(3), 256-279.

[5] Li, X., "A Non-dominated Sorting Particle Swarm Optimizer for Multiobjective Optimization", in Proceeding of Genetic and Evolutionary Computation Conference 2003 (GECCO'03), Chicago, USA, 2003.

[6] Raquel R.R , Naval P.C. , "An effective use of crowding distance in multiobjective particle swarm optimization", in Proceeding of Genetic and Evolutionary Computation Conference 2005 (GECCO'05), Washington DC, USA, 2005.

[7] Fieldsend, J. and Singh, S., "A multi-objective algorithm based upon particle swarm optimization, an efficient data structure and turbulence", in Proc. 2002 U.K. Workshop on Computational Intelligence, Birmingham, UK., 2002, pp.37-44.

[8] Deb K., Agrawal S., Pratap A., Meyarivan T., "A fast elitist non-dominated sorting genetic algorithm for multi-objective optimisation: NSGA-II", IEEE Transactions on Evolutionary Computation, 2002, 6(2), 182-197.

[9] Whitley, D., "The genitor algorithm and selection pressure: why rank-based allocation of reproductive trials is best", In: Proceedings of the Third International Conference on Genetic AlgorithmsArlington, VA, Morgan Kaufman Publishers, San Mateo, CA, 1989, pp. 116–121.

[10] Eberhart R., Shi Y., "Comparing inertia weights and constriction factors in particle swarm optimization", In: Proceedings of the 2000 Congress on Evolutionary Computation. Washington, DC, 2000, pp.84-88.

[11] Knowles J., Corne D., "On metrics for comparing non-dominated Sets", In Congress on Evolutionary Computation (CEC 2002), 2002.

[12] "NSGA-II: a multi-objective optimization algorithm", http://www.mathworks.com, Available on Jun 2006.

[13]"Particle Swarm Optimizer", http://www.particleswarm.info/Programs.html, Available on Jun 2006.

[14] Schaffer J.D., "Multiple Objective Optimisation with Vector Evaluated Genetic Algorithms", Proceedings of the 1st International Conference on Genetic Algorithms,1987.

[15] Fonseca C.M., Fleming P.J., "Multi-objective optimization and multiple constraint handling with evolutionary algorithms-Part II: Application example", IEEE Transactions on Systems, Man, and Cybernetics: Part A: systems and Humans, 1996, pp.38-47.