A Comparison of Messy GA and Permutation based GA for Job Shop Scheduling

Pio Fenton
Cork Institute of Technology
Cork
Ireland
+353863814019
pfenton@cit.ie

Paul Walsh
Cork Institute of Technology
Cork
Ireland
+353871348353
pwalsh@cit.ie

ABSTRACT
This paper presents the results of a fair comparison between a messy GA and a permutation based simple GA as applied to a job shop scheduling system. An examination is made at a macro level in terms of performance and quality of schedules achieved and conclusions are drawn as to the superiority of messy GA or otherwise.

Categories and Subject Descriptors
I.2 [**Artificial Intelligence**]

General Terms
Algorithms.

Keywords
Messy Genetic Algorithms, Repeating Permutation Representation, Job Shop Scheduling.

1. RESULTS
The following results for the permutation-based system are achieved using an implementation of the GALIB library [1], which had to be extended greatly to handle permutations with repetition. Parameters used are those identified as most suitable by Mattfeld [2]. Likewise for the messy GA we implemented the source code made available by [3]. This too had to be extended and parameters used are those similar to those, which have been used in the OMEGA system albeit for a different type of scheduling problem. Table 1 lists these parameters. All tests involve 50 runs on identical machines running Windows 2000 and with a Pentium platform.

We examine this problem for two fronts. First we examine the performance in terms of makespan for 2 benchmark problems: ft10 [15] and a 15 x 15 [16] instance from the Taillard benchmark sets, given equal population sizing parameters. We then examine the performance of the systems given equal processing times. For the FT10 benchmark we test both simple and messy GA with population sizes 100 & 200 running over 200 generations (Figure 1 & 2). Other parameters are as in Table 1. Here we see that simple GA outperforms mGA in terms of makespan achieved. The optimum for this problem is 930, and while simple GA with active scheduling often attains scores about 950, we see that mGA never scores higher than 990.

Table 1 Parameters used in each system

Parameter	Simple GA	Messy GA
Pop. Size	Varies	Varies
Generations	200	25
Crossover	GOX 0.6	Splice (0.01)
Mutation	PBM 0.03	0.0
Era	--------	4
Epoch	--------	2
Elitism	Yes	Yes

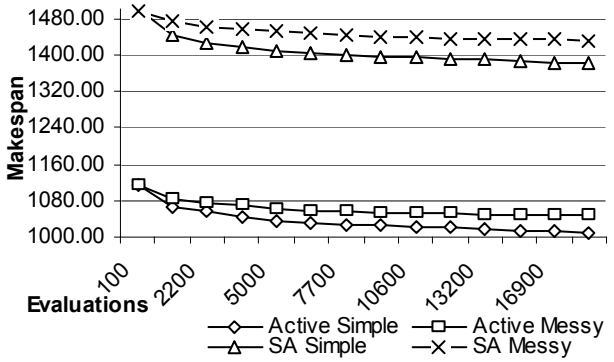

Figure 1 Makespan achieved for FT10 (population size 100)

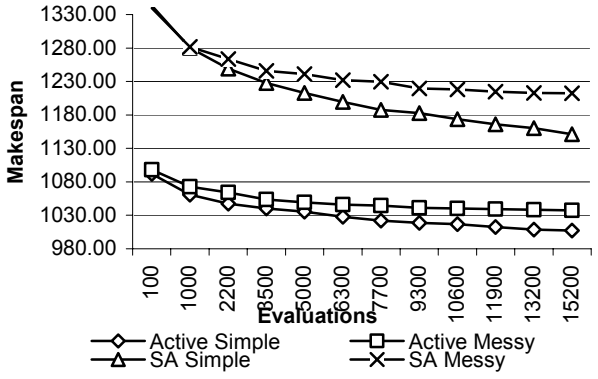

Figure 2 Makespan achieved for FT10 (population size 200)

GECCO'05, June 25-29, 2005, Washington, DC, USA.
ACM 1-59593-010-8/05/0006.

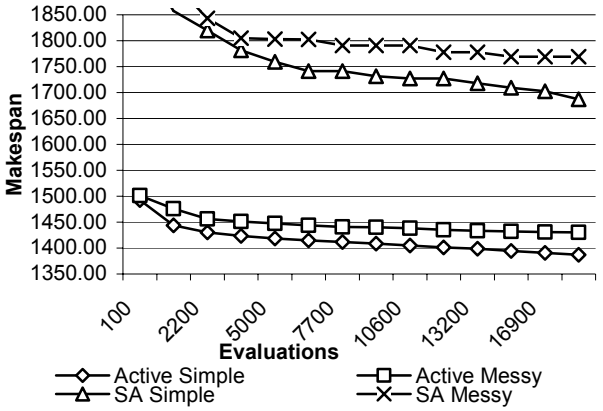

Figure 3 Average makespan achieved on a Taillard problem averaged over 50 runs with population size 100.

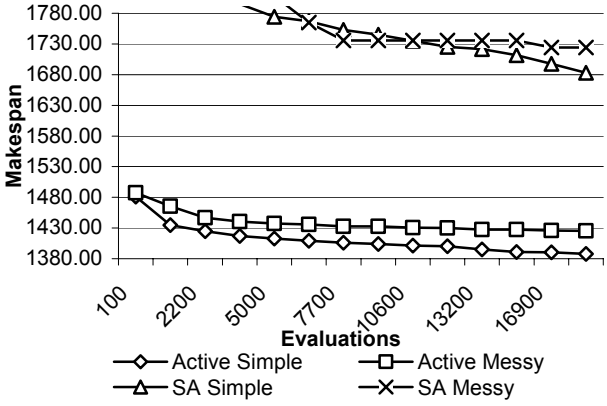

Figure 4 Average makespan achieved on a Taillard problem averaged over 50 runs with population size 200.

Figures 8, 9 0 highlight the performance of both GA systems with the Taillard benchmark problem. The best-known makespan for this is 1251. Here active scheduling in the simple GA outperforms mGA consistently.

The t-Test results for the above experiments are presented in Tables 4 and 5, and they support the conclusions presented in this section.

Table 2 t-Test results for comparisons of Messy and Simple GA systems using an active schedule builder.

Comparison	2-tailed P-Value
FT10 (pop 100)	7.87E-164
FT10 (pop200)	4.173E-15
Taillard (pop 100)	4.108E-37
Taillard (pop 200)	2.102E-37

Given that the CPU time is less in messy GA than in simple GA, we present the results of tests focusing on whether a Messy GA given equal CPU time as a simple GA, can perform as well as the latter. We find that though there is improvement with mGA, the simple GA outperforms the messy GA consistently. Figure 15 highlights the performance of mGA and the simple GA given equal processing time. The difference between makespans achieved narrows only slightly.

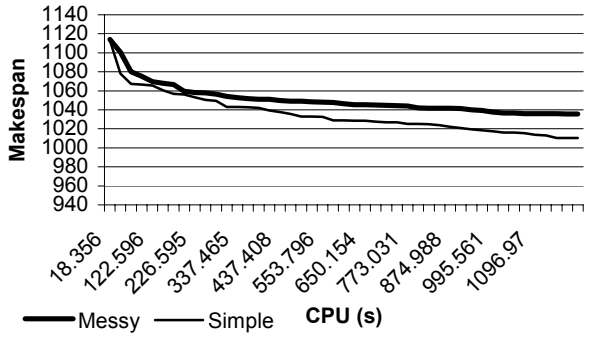

Figure 5 mGA and simple GA given equal processing time with population size 100

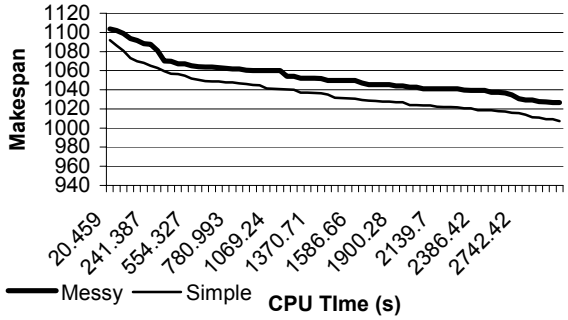

Figure 6 mGA and Simple GA given equal processing time with population size 200

2. CONCLUSIONS

From the above comparisons a number of findings are apparent. While the messy GA performs better in terms of computation time, the quality of its makespans are not as good as those in the permutation based GA. When given equal CPU time mGA does not improve correspondingly. Also the results indicate that the simple GA scales up better in terms of its performance. It seems that in terms of job-shop scheduling, that the advances made in overcoming the linkage problem with mGA, are not in themselves sufficient in helping to tackle JSSP, and that with more difficult problems, some further issues have arisen which have limited its ability.

References

1. www.lancet.mit.edu/ga (accessed 10-January-2005)
2. Dirk Christian Mattfeld, "Evolutionary Search and the Job-Shop: Investigations on Genetic Algorithms for Production Scheduling"1995 Spinger- Verlag
3. Dimitri Knjazew "OmeGa: A competent Genetic Algorithm for Solving Permutation and Scheduling Problems" 2001, Kluwer Academic Press.
4. Muth, J. F. and Thompson, G. L., eds., *Industrial Scheduling*, Englewood Cliffs, N. J.: Prentice-Hall, Inc., 1963.
5. ta01-ta80 are from É. D. Taillard (1993), "Benchmarks for basic scheduling problems*", European Journal of Operational Research* 64, Pages 278-28