International Journal of Applied Engineering Research
ISSN 0973-4562 Volume 10, Number 2 (2015) pp. 4589-4596

http://www.ripublication.com

Novel Memetic Algorithms for Flexible Manufacturing Systems

F. Choong*

School of Engineering, Taylor's University, No. 1 Jalan Taylor's
47500 Subang Jaya, Selangor DE, Malaysia
**Corresponding Author: FlorenceChiaoMei.Choong@taylors.edu.my*

S. Phon-Amnuaisuk

Faculty of Business and Computing, Institut Teknologi Brunei, Mukim Gadong A,
BE1410 Brunei
somnuk.amnuaisuk@itb.edu.bn

M.Y. Alias
Faculty of Engineering, Multimedia University, 63100 Cyberjaya, Malaysia
yusoff@mmu.edu.my

Abstract

Flexible manufacturing systems (FMS) are systems composed of multiple heterogeneous machines and to obtain optimal schedules or solutions to FMS problems is a complex task. Evolutionary algorithms have been a popular approach to finding schedules for FMS. These algorithms, while effective, are dependent on the quality of initial populations and may not converge completely to a global optimum. This paper presents two novel memetic algorithms that combine adaptive Genetic Algorithm (AGA) with simulated annealing (SA) and local search (LS). These search techniques are used to initialize the chromosome population, enhance convergence, and refine the final schedule in GA. The resulting memetic algorithms are compared against each other and against traditional techniques (GA, SA and LS). Experimental results reveal that these memetic techniques have effectively produce improved solutions over conventional methods, often with faster convergence.

Keywords: Flexible manufacturing system, genetic algorithm, adaptive genetic algorithm, simulated annealing, local search, memetic algorithm.

1. Introduction

Flexible manufacturing systems (FMS) are systems composed of multiple heterogeneous machines. Each machine is capable of performing multiple operations. Work is performed on parts, which each require a different ordered series of operations to be performed by the machines. FMS are characterized by overlap between machine capabilities, which allows parts to take multiple potential paths through the machines. This flexibility allows for many possible schedules for the parts, but introduces the need to identify the schedules that are most beneficial.

To decide which schedule is best, multiple objectives may be considered. These objectives may include minimizing part transfer (moving parts between machines), balancing load on machines (keeping all machines running instead of some being idle), and minimizing the type of operations performed by each machine (avoiding the need for retooling between operations). For the purposes of this work, only the first two objectives listed here are considered.

Optimal schedules, or solutions, to FMS problems cannot be found using polynomial time algorithms. Metaheuristic approaches have frequently been adopted when dealing with such problems. A popular approach has been to use evolutionary or genetic algorithms (GAs). GAs are capable of searching a wide range of the solution space, avoiding being trapped by local optima. In addition, they are useful when multiple scheduling objectives are considered. Chen and Ho (2001) successfully used a GA to generate an entire range of Pareto-optimal solutions to a multi-objective FMS problem in a single run.

Recent work has led to the exploration of hybridized genetic approaches, or 'memetic' approaches, to improve convergence and address some of the limitations of evolutionary algorithms (Choong et al, 2011). Crossover and mutation operators in GAs may be useful for exploring the global solution space, which may permit the finding of a 'good' solution close to the optimal (Choong et al, 2009a). However, GAs are not well suited to performing localized hill climbing.

Previous work has shown success using hybridized evolutionary algorithms combining adaptive GAs with a local search (LS) (Choong et al, 2009b). Younes et al. (2009) employed a simple local search to 'polish' the best single individual from the population. The assumption made is that the best individual(s) produced by the GA is near the optimal solution, and a localized improving search will bring the found solution closer to it.

Essafi et al. (2008) employed LS to improve solutions produced by crossover operators at each generation. An iterative LS, which allows some non-improving perturbations, was found to have better performance than a direct steepest descent search

in this hybridized solution. This suggests that more exploratory heuristics, in addition to hill climbing heuristics, should be investigated in hybridized GAs.

Simulated annealing (SA) is another technique employed independently in FMS scheduling (Kim & Kim, 1996). SA is also a global optimization technique that initially performs a large amount of exploration by permitting some non-improving moves on a single schedule, and slowly converges to some optima by reducing the likelihood of a non-improving change. Under some circumstances, SA can generate

better schedules than genetic algorithms for similar scheduling problems (Suppapitnam et al, 2000). In addition to being used to develop schedules independently, SA has shown promise in combination with other global optimization techniques, such as particle swarm optimization (PSO) (Low et al, 2004) and GAs (Kumar & Rockett, 2009).

One limitation of GAs is the sensitivity of the algorithm to the initial population. Randomly generated chromosomes tend to result in poorer performance as compared to when the initial population is developed through some other heuristic (Choong et al, 2009b; Wei & Wu, 2005). Hybridized methods where the initial population is improved prior to evolution should be investigated.

This paper explores hybrid combinations of adaptive GAs with either LS or SA techniques to solve this problem. Furthermore, the placement of LS and SA within the GA (to initialize the population, to 'polish' a final result, or to improve the convergence at each generation) is studied. These various options are evaluated and compared to assess their relative merits. The findings show that by carefully applying either local search or SA in an adaptive GA play an important role in arriving at a near optimal solution in reasonable time and in escaping from entrapment in local optima. Consequently, a heuristic-supported-implementation of the GA algorithm provides better alternatives for the solution of optimization problems (Choong et al, 2009a).

This work serves as an initial investigation for the suitability of applying adaptive approaches and memetic algorithms to solve optimization problems associated with scheduling in FMS systems. It also aims to address the many short comings from current techniques reported in the literature. As a summary, this work contributes in the following areas:

1. Two new memetic algorithms that incorporate simulated annealing (AGA-SA) and local search (AGA-LA) into the adaptive GA. The adaptive GA used is based on the previous work (Choong et al, 2008; Choong et al, 2009a). It is meant to prevent GA from getting stuck in local optima at an early stage and non-improvement during remaining iterations and to improve its performance and solution quality. These integration of LS and SA into the adaptive GA is based on the following three scenarios:
   a) To improve convergence after each generation of the adaptive GA
   b) To initialize the population and provide the adaptive GA with a better starting point
   c) To enhance the final result of the adaptive GA
2. The proposed method demonstrated superiority in performance when compared to simulated annealing, GA and particle swarm optimization. It can be used as foundation for future work in FMS and can be easily be adapted to solve other types of scheduling and optimization problems.

The remainder of this chapter is laid out as follows. Section 2 discusses the variety of hybridized techniques explored, while Section 3 gives specific implementation details for the LS and SA algorithms. Section 4 describes parameter settings for each of the techniques, benchmark problems, and experimental results. Discussion of

results is given in Section 5. The paper ends with conclusions about the hybrid techniques investigated and possible future work.

2. A Memetic Approach

Both LS and SA are considered as candidates for hybridization with an adaptive GA. The resulting techniques are referred to as memetic algorithms. In addition, three potential locations of both techniques within the memetic algorithms are explored.

Both LS and SA are search techniques that make incremental changes to a single schedule. With LS, only improving changes are accepted–if the fitness of the schedule is decreased with the change, that change is rejected. SA allows some non-improving changes to the chromosome to be accepted. The probability of accepting a non-improving change decreases with time, meaning that the solution should converge towards some optimum, without being constrained to the local optimum that the solution from the GA may be in.

The second aspect of the approach is a comparison of the placement of the LS or SA within the GA. A typical GA is shown in Fig. 1. The first option is to perform a search on the initial population of the GA before the generations begin, as shown in Fig. 2. This will provide the GA with a better starting point, though it may also reduce the diversity of the population and cause the population to become trapped in a local optimum. The second option is to use the GA first and apply the search technique to the final population to polish the final results, as shown in Fig. 3. It is only necessary to apply the search technique to the best members of the population.

```
initialize population of schedules randomly
for number of generations {
   mate a selected percentage of schedules
      replace parent schedules with children
   mutate a selected percentage of schedules
}
return best schedule
```

Fig. 1. A simplified genetic algorithm

```
initialize population of schedules randomly
improve a subset of the schedules through local
search or simulated annealing
for number of generations {
   sort schedules by fitness
   mate a selected percentage of schedules
      replace parent schedules with children
   mutate a selected percentage of schedules
}
return best schedule
```

Fig. 2. Hybrid genetic algorithm with population initialization

```
initialize population of schedules randomly
for number of generations {
   sort schedules by fitness
   mate a selected percentage of schedules
      replace parent schedules with children
   mutate a selected percentage of schedules
}
improve best schedule through local search or
simulated annealing
return best schedule
```

Fig. 3. Hybrid genetic algorithm with final result optimization

The third and final option is to apply the search after each generation of the GA, as shown in Fig. 4. This will help to increase the convergence rate of the GA, but this must be done carefully to prevent premature convergence. The LS or SA will need to be applied to only a few of the fittest members of the population. This differs from some previous hybrid approaches. Ishibuchi and Murata (1998) developed a genetic local search algorithm where all members of the population were improved through local search in every generation. Similarly, Wang and Zheng (2001) applied SA on the entire population for every generation of a GA. In this research, applying a LS or SA operator to every member of the population was found to be extremely computationally expensive, so only a subset of the population is selected for improvement in each generation.

```
initialize population of schedules randomly
for number of generations {
   sort schedules by fitness
   mate a selected percentage of schedules
      replace parent schedules with children
   mutate a selected percentage of schedules
   improve a subset of the schedules through
   local search or simulated annealing
}
return best schedule
```

Fig. 4. Hybrid genetic algorithm with search at every generation

3. Implementation

The adaptive GA used for solving static FMS problems was based on our previous work (Choong et al, 2008; Choong et al, 2009a). It is an adaptive GA that automatically adapts the probabilities of crossover and mutation to realize the twin goals of maintaining diversity in the population and sustaining the convergence capacity of the GA. The adaptive GA overcomes the inherent drawbacks of the standard GA. In this work, LS and SA were added into the adaptive GA in three different placements. First, LS was added to the adaptive GA with three possible placements. These placements are on the initial population of the GA (LS-Before-GA), on the final generation of the GA (LS-After-GA), and after each generation in

the GA (LS-Throughout-GA). The LS was also set up to run by itself without the GA for comparison (LS-Only). The LS implementation is shown in the pseudo-code of Fig. 5. The loop continues until no change has been accepted in the last *x* iterations, where *x* is a parameter set independently for each placement.

```
do {
   find some schedule change in neighbourhood
   delta = newschd.fitness - oldschd.fitness
   if(delta > 0)
      accept change
   else
      reject change
} while(change accepted in last x iterations)
```

Fig. 5. A Local Search algorithm for FMS

Second, SA was implemented and added in the same three placements (SA-Before-GA, SA-After-GA, SA-Throughout-GA). SA can also be run by itself without the GA (SA-Only). The SA implementation is shown in the pseudo-code of Fig. 6. The cooling rate, initial temperature, and final temperature are all parameters set independently for each placement.

```
do {
   do {
      find some schedule change in neighbourhood
      delta = newschd.fitness - oldschd.fitness
      if(delta > 0)
         accept change
      else if(random < exp(delta/temperature)
         accept change
      else
         reject change
   } while(changes happening)
   temperature = cooling rate * temperature
} while(temperature > final temperature)
```

Fig. 6. A Simulated Annealing algorithm for FMS

4. Evaluation

Test Parameters

The genetic parameters used by all test cases are listed in Table 1. These parameters were not varied between test cases. Parameters for SA were varied between memetic techniques. SA requires an initial temperature, cooling rate, and final temperature point. In addition, the subset of the genetic population undergoing SA is varied for each meta-heuristic. Trial and error was performed to determine a favorable set of parameters for each method. These parameters are listed in Table 2. Two parameters can be varied for local search: the number of unsuccessful incremental changes to the schedule before the algorithm and size of the subset of the genetic population undergoing LS. Both parameters are listed in Table 3.

Table 1 Genetic Parameters

Parameter	Value
Number of runs	50
Population size	400
Number of generations	100
Population initialization	Random
Mutation rate	5%
Mutation method	Simple
Crossover rate	99%
Crossover method	Uniform
Selection method	Simple sort
Replacement method	Replace most inferior parent
Objective function	Fixed weights

Table 2 Simulated Annealing Parameters

Technique	T_0 (10^6)	T_f (10^6)	Cooling Rate	Subset of Population
SA-only	5000	10	0.999	N/A
SA-Before-GA	1000	100	0.990	5%, random
SA-After-GA	10	1	0.990	Best solution
SA-Throughout-GA	1000	100	0.500	5%, random

Table 3 Local Search Parameters

Technique	Iterations w/o change	Subset of population
LS only	100	N/A
LS before GA	30	10%, random
LS after GA	50	Best solution
LS throughout GA	20	5%, random

Benchmark Cases

Each of the meta-heuristic techniques was applied to a series of benchmark test cases with varying number of parts, operations, and machines. A summary of the benchmarks used is found in Table 4.

Table 4 Benchmark Data

Benchmark	Machines	Parts	Operations
1	3	1	5
2	3	2	5
3	3	3	5
4	5	2	7
5	5	5	6
6	5	10	8
7	6	6	10
8	7	7	7
9	10	15	9
10	11	20	9
11	13	28	12
12	16	34	14
13	20	42	16

4.3 Metrics

The average fitness of the best schedule over 50 test runs is used to measure the overall performance of each algorithm for each benchmark. This fitness value is a weighted aggregate of the two objectives considered in this paper: part transfer and machine load balance.

Processing time for each algorithm is also a significant consideration. If schedules need to be developed rapidly, improved optimization performance may need to be sacrificed in the interest of time. Note that convergence over the number of generations alone cannot be used as a basis for fair comparison between the techniques considered in this paper. The reason is simple: one generation of a memetic algorithm is considerably longer than that of a pure GA. Hence, convergence over time should be also considered.

Results

Each algorithm was tested 50 times using the 13 sets of benchmark data. A MacBook Pro with a 2.4 GHz dual-core CPU was used for testing. The fitness of the best schedule from each run, as well as the computation time required to perform each method were recorded. Average combined fitness functions and computation times are listed for each benchmark and algorithm in Table 5. For the benchmark cases with fewer machines, performance of the genetic algorithms and hybridized techniques were approximately the same. Pure SA and LS were found to perform the worst on average. Fig. 7 shows performance for the first benchmark case, with 3 machines, 1 part, and 5 operators.

Table 5 Summary of Average Fitness Values and Computation Times

Benchmark No.		GA-Only	SA-Before-GA	SA-After-GA	SA-Thrght-GA	SA-Only	LS-Before-GA	LS-After-BA	LS-thrght-GA	LS-Only
1	Fitness	0.58	0.58	0.58	0.58	0.54	0.58	0.58	0.58	0.54
	(Time)	(0.23)	(0.26)	(0.23)	(0.34)	(0.16)	(0.22)	(0.23)	(0.29)	(0.00)
2	Fitness	0.59	0.59	0.59	0.59	0.59	0.59	0.59	0.59	0.58
	(Time)	(0.25)	(0.28)	(0.26)	(0.35)	(0.14)	(0.26)	(0.25)	(0.33)	(0.00)
3	Fitness	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50
	(Time)	(0.25)	(0.26)	(0.25)	(0.31)	(0.11)	(0.25)	(0.25)	(0.32)	(0.00)
4	Fitness	0.65	0.65	0.65	0.65	0.64	0.65	0.65	0.65	0.63
	(Time)	(0.34)	(0.43)	(0.34)	(0.50)	(0.18)	(0.37)	(0.34)	(0.49)	(0.00)
5	Fitness	0.67	0.67	0.67	0.67	0.67	0.67	0.67	0.67	0.66
	(Time)	(0.25)	(0.27)	(0.25)	(0.33)	(0.13)	(0.25)	(0.25)	(0.32)	(0.00)
6	Fitness	0.82	0.83	0.82	0.83	0.80	0.83	0.82	0.83	0.77
	(Time)	(0.25)	(0.29)	(0.26)	(0.32)	(0.13)	(0.27)	(0.25)	(0.37)	(0.00)
7	Fitness	0.88	0.88	0.88	0.89	0.85	0.88	0.88	0.89	0.82
	(Time)	(0.28)	(0.30)	(0.28)	(0.35)	(0.13)	(0.28)	(0.28)	(0.38)	(0.00)
8	Fitness	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
	(Time)	(0.08)	(0.09)	(0.08)	(0.11)	(0.12)	(0.08)	(0.08)	(0.13)	(0.00)
9	Fitness	0.74	0.79	0.78	0.78	0.80	0.77	0.76	0.80	0.75
	(Time)	(0.63)	(0.66)	(0.63)	(0.77)	(0.18)	(0.64)	(0.63)	(0.98)	(0.00)
10	Fitness	0.74	0.76	0.76	0.77	0.76	0.76	0.75	0.77	0.70
	(Time)	(0.55)	(0.60)	(0.56)	(0.70)	(0.18)	(0.58)	(0.55)	(0.88)	(0.00)
11	Fitness	0.70	0.73	0.72	0.73	0.73	0.72	0.72	0.74	0.67
	(Time)	(0.75)	(0.84)	(0.76)	(0.97)	(0.22)	(0.80)	(0.76)	(1.28)	(0.00)
12	Fitness	0.67	0.72	0.70	0.72	0.74	0.69	0.69	0.72	0.64
	(Time)	(0.92)	(1.05)	(0.93)	(1.23)	(0.30)	(1.00)	(0.92)	(1.68)	(0.01)
13	Fitness	0.66	0.70	0.70	0.70	0.73	0.68	0.70	0.72	0.66
	(Time)	(1.21)	(1.41)	(1.23)	(1.68)	(0.41)	(1.36)	(1.22)	(2.42)	(0.01)

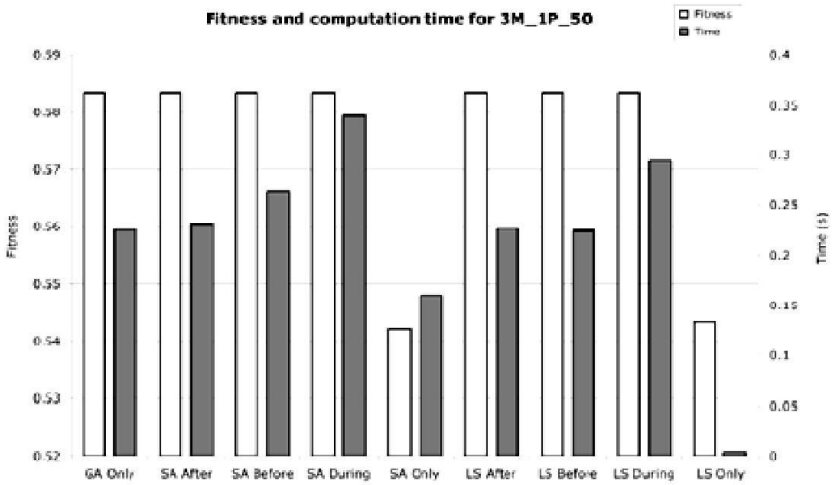

Fig. 7. Average fitness function values for 3 Machines, 1 Part, and 5 Operators (Benchmark #1)

As the size and complexity of the benchmark data is increased, the hybridized techniques were all found to offer better performance than the genetic approach alone,

as shown in Fig. 8 and Fig. 9. The best average fitness performance among the memetic algorithms was given by LS throughout GA and SA throughout GA.

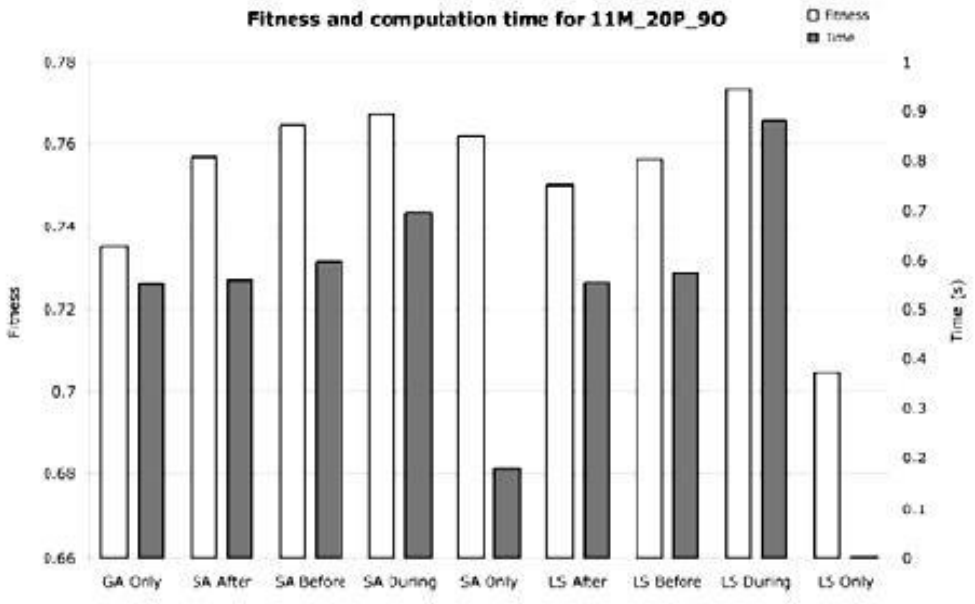

Fig. 8. Average fitness function values for 11 Machines, 20 Parts, and 9 Operators (Benchmark #10)

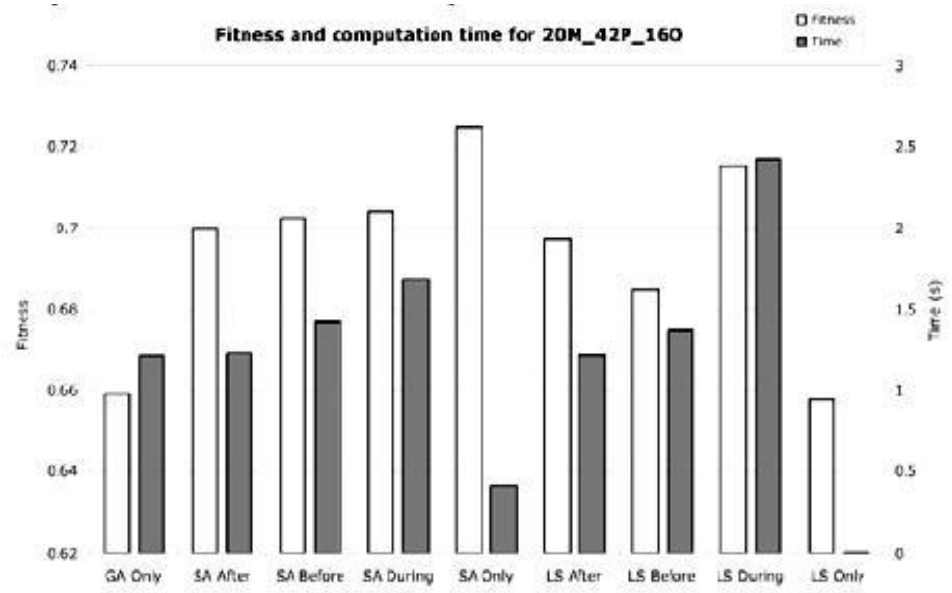

Fig. 9. Average fitness function values for 20 Machines, 42 Parts, and 16 Operators (Benchmark #13)

SA-Before-GA outperformed SA-After-GA; LS-Before-GA resulted in a significantly lower average fitness than LS-After-GA. With the exception of LS-Throughout-GA, the SA-hybrids outperformed their LS counterparts in terms of fitness.

The average fitness from SA-Only increased relative to the fitness of other algorithms as the benchmark size increased. For the largest benchmark cases (Fig. 9), SA-Only exhibited the best average fitness.

As would be expected, larger benchmarks required more computation time than smaller ones. However, the pattern of time performance for each of the algorithms was generally the same for all the benchmarks (Fig. 7, Fig. 8, and Fig. 9). Since the memetic approaches add operations to the genetic algorithm, these approaches consume more time than GA-only. For both the hybrid LS and SA algorithms, the least time is consumed by the 'After' cases, as only one schedule is optimized, as opposed to a larger subset for 'Before'. The 'Throughout' cases require the most processing time, as optimization must be performed on a subset every generation. Though it exhibited the best performance in terms of fitness among the memetic algorithms, the processing time cost of LS-Throughout-GA was significant – as much as twice the time consumed by GA-Only was required for the largest benchmark case.

SA-Only and LS-Only required significantly less processing time, as they did not include any genetic processing and were only performed using a single schedule for each run. LS-Only required relatively no time at all, as it stops when it cannot find any more improving schedule perturbations. Convergence patterns were roughly the same for all of the benchmarks. A plot of the best fitness values over time for the largest benchmark case is shown in Fig. 10. The convergence of every algorithm is shown, except for LS-Only, which had low performance and negligible computation time.

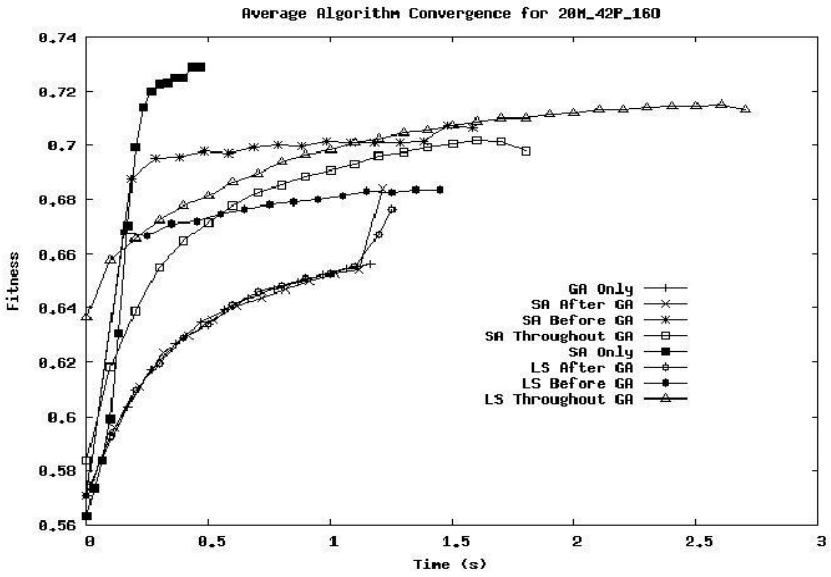

Fig. 10. Best fitness over time for 20 Machines, 42 Parts, and 16 Operators (Benchmark #13)

For LS-After-GA and SA-After-GA, the convergence curve is identical to the GA-Only curve, except at the very end, where both curves rise rapidly with the LS/SA step. LS-Before-GA and SA-Before-GA both converge rapidly at first, but the rise in fitness slows quickly, and the curves level off. LS-Throughout-GA and SA-Throughout-GA converge more rapidly than GA-Only, but take a longer amount of time to plateau. SA-Only is shown to have very rapid convergence.

5. Discussion

For the benchmark cases with 5 or fewer machines, performance was roughly the same for all memetic algorithms. With a small number of machines, the genetic-based algorithms should be able to explore a greater part of the solution space. Since GA-Only is simpler and less time consuming than the hybrid solutions, it should be more suitable for use with small FMS problems.

SA-Only performed similar to LS-Only with the smaller benchmarks. This was unexpected, as SA is capable of exploring regions outside the local minima. However, with the smaller benchmarks, incremental changes have more significant effect on the overall fitness. If the SA algorithm were caught in some local minima, a non-improving change would have a significantly lower likelihood of acceptance than with larger benchmark data.

As expected, LS-Only exhibited very poor performance relative to all other algorithms, since LS has no mechanism for escaping local minima. Thus performance depends entirely on the initial schedule given to the algorithm. For most of the benchmarks with more than 5 machines, the 'Before' algorithms outperformed the corresponding 'After' algorithms, confirming the assumption that a good initial population can assist an optimization routine. However, for the largest set of benchmark data, where LS-Before-GA resulted in a significantly lower average fitness than LS-After-GA, it is conceivable that optimization of the initial schedule in this case may have resulted in decreased diversity between schedules. From the convergence graph, it is clear that the 'Before' algorithms had rapid initial convergence, followed by slow, level improvement. This suggests that most of the optimization was taking place in the initialization phase and that the genetic algorithm was able to provide little improvement, as population diversity had been reduced. Thus, reasonably equivalent performance could be achieved using fewer generations.

Since the search technique is only applied at the end in the SA-After-GA and LS-After-GA approaches, no improvement is seen in the convergence of the algorithms as compared to GA-Only until evolution is complete. Further investigation should be done to determine if similar performance could be achieved with fewer generations.

The LS-Throughout-GA and SA-Throughout-GA approaches both exhibited improved convergence rates over GA-Only. These methods took significantly longer to complete the full set of evolutionary generations in the tests than the other hybrid approaches. However, they were capable of reaching much better fitness values than other hybrid approaches within the same amount of time as the other algorithms. The LS-based method can be seen to provide the best performance within a fixed time of any memetic approach.

SA average fitness increased with benchmark complexity. For the largest benchmark case, SA-Only outperformed every other method, in the least amount of time (aside from LS-Only). It would normally be expected of simulated annealing algorithms to require more processing time than a well-tuned genetic algorithm, so its favorable performance here is unexpected. This suggests that the genetic parameters are not well suited to this particular set of benchmark data. Further study should investigate the impact that varying genetic parameters, such as crossover rates and operators, mutation rates, population size, etc., has on the hybridized algorithms. In addition, since the LS and SA parameters were determined through trial and error, improved parameters should be determined for each meta-heuristic in a more thorough study.

6. Comparison with Related Work

The first comparison was performed between the proposed hybrid GA (AGA-SA) with another GA implementation by Kumar and Rockett (2009) and PSO by Ponnambalam and Low (2008). The PSO employed a local search method called Random Start Adjacent Swapping Scheme (RSASS). Fig. 11 illustrates the computation time for the three algorithms. It can be clearly seen that the proposed method consumed the least time in many instances, however, the increase in the complexity of the problem increases the computation time as oppose to GA and for 1,024 tasks, GA consumes the smallest computation time. Next, the second proposed hybrid GA (AGA-LS) combined with local search after each generation was also compared with three other algorithms reported in the literature. The first two algorithms are multiobjective simulated annealing implementation using different method of finding the acceptance possibility of a neighbourhood solution. The first algorithm employs the calculation given by Suppapitnarm et al. (2000) referred to SMOSA, and the next algorithm takes the calculation by Ulungu et al. (1999), referred to as UMOSA. The third algorithm is using GA and comparisons were performed against a reference set. Figures 12 and 13 present the distributions of the reference solutions and performance of the proposed hybrid algorithm, GA, SMOSA and UMOSA for two benchmark instances. The proposed algorithm effectively obtained results that are closer to the reference set. The results indicate that the proposed algorithm significantly outperforms the other three algorithms.

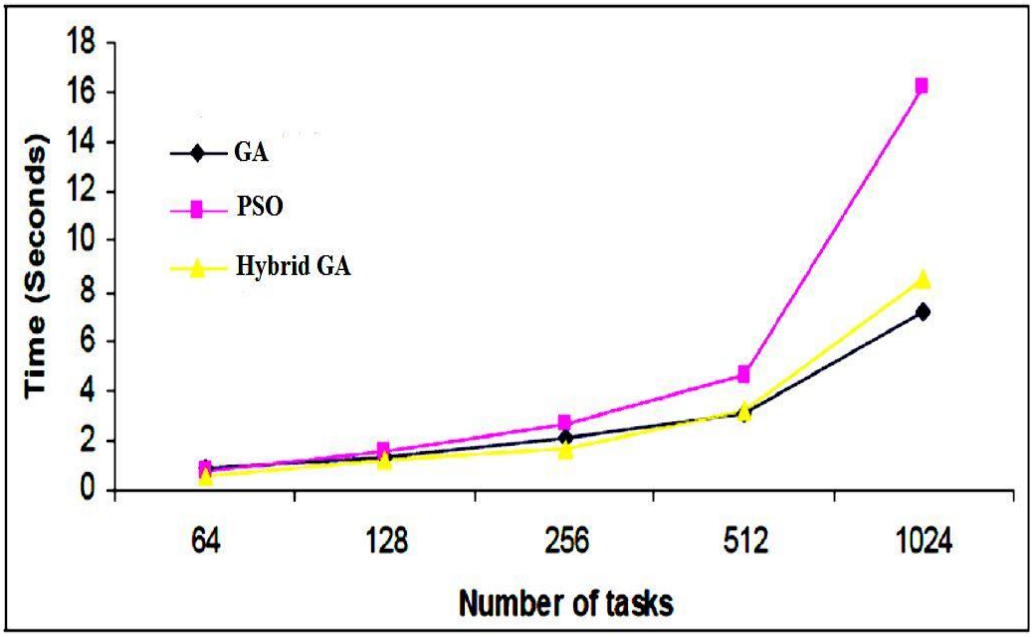

Figure 11 Performance of the algorithms.

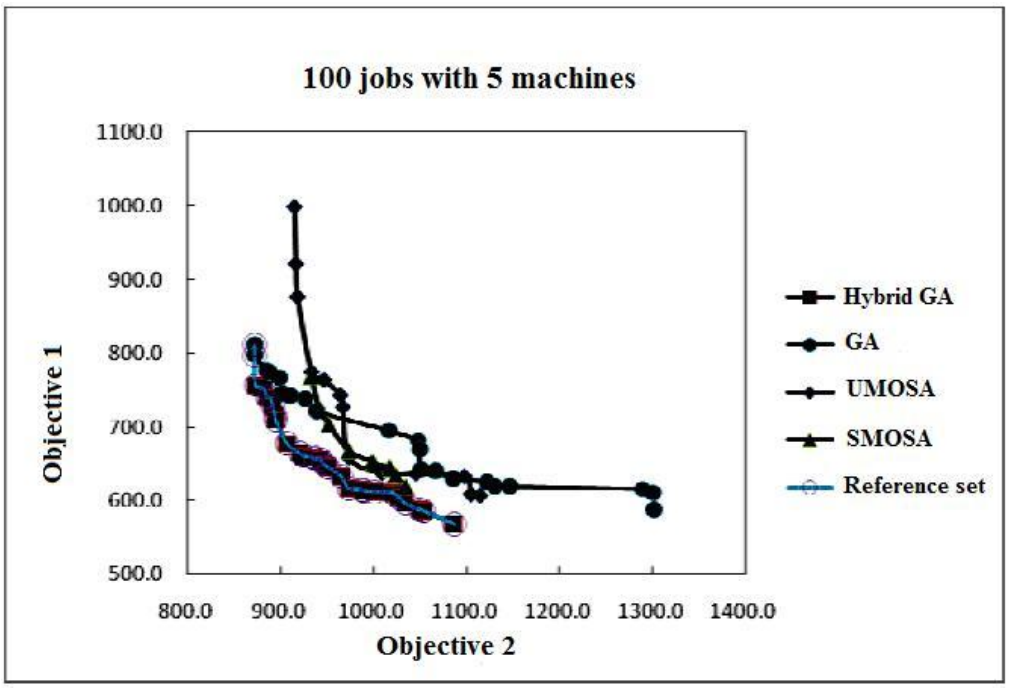

Figure 12 Comparison of the performances of four algorithms with (100 jobs, 5 machines)

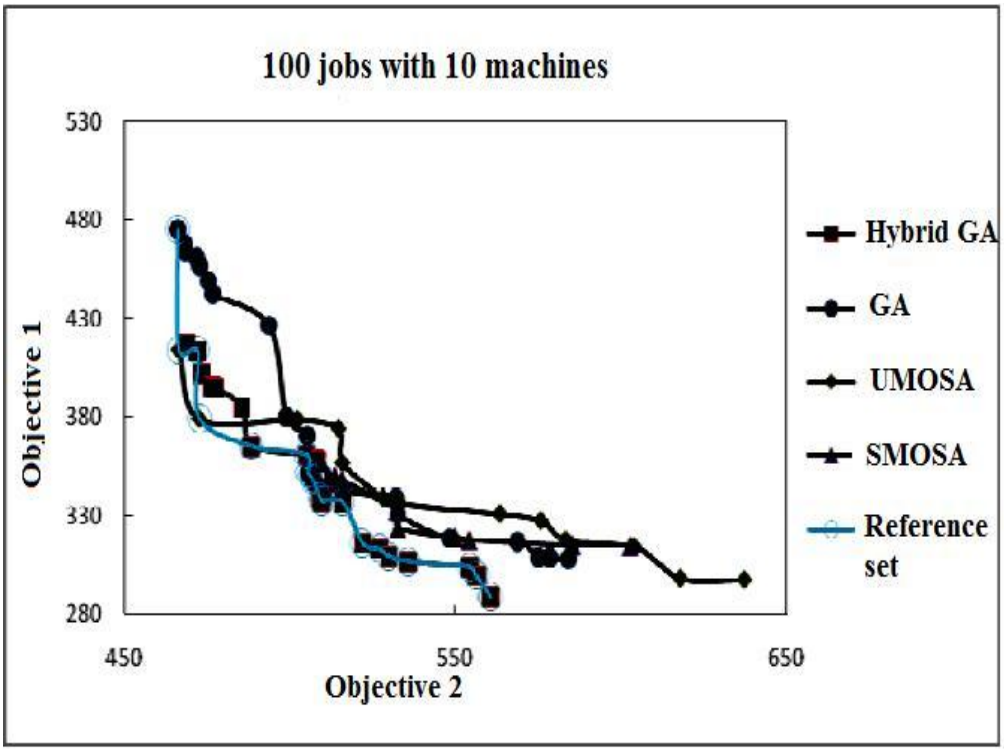

Figure 13 Comparison of the performances of four algorithms (100 jobs, 10 machines)

7. Conclusion

This paper presented several hybridized genetic techniques for solving FMS scheduling problems. These involved the use of LS and SA to improve convergence, initialize the population, or enhance a final result for an adaptive GA. Results obtained indicate that these memetic algorithms can offer improved performance over traditional GAs for larger FMS problems. Though more time consuming for the same number of generations, these algorithms could potentially be used with fewer generations and still offer better performance. It was determined that improving a subset of each generation through LS in an adaptive GA resulted in the best convergence for all of the algorithms investigated. All memetic approaches were compared using the same adaptive GA and parameters. Therefore, further study into the impact of these parameters on the memetic algorithms is recommended.

8. References

1. A.Younes, S. Areibi, and P. Calamai, A hybridized evolutionary algorithm for flexible manufacturing systems in dynamic environments, The Third

International Conferences on Modeling, Simulation and Applied Optimization (ICMSAO'09), The American University, Sharjah, UAE. (2009).
2. A.Suppapitnarm, K.A. Seffen, G.T. Parks, and P.J. Clarkson, Simulated annealing: an alternative approach to true multi-objective optimization, Engineering Optimization. 33(12) (2000) 59-85.
3. C. Low, J. Y. Yeh, and K. I. Huang, A robust simulated annealing technique for flow shop scheduling problems, International Journal of Advanced Manufacturing Technology. 23(5) (2004) 762-767.
4. F. Choong, S. Phon-Amnuaisuk, and M.Y. Alias, Adaptive Genetic Algorithm: An Essential Ingredient in High-Level Synthesis, Proceedings of the IEEE Congress on Evolutionary Computation (CEC), June 1-6, Hong Kong, China. (2008) 3837-3844.
5. F. Choong, S. Phon-Amnuaisuk, and M.Y.Alias, New Operators of GA for Improving the Performance of High-Level Synthesis, International Journal of Computational Intelligence and Research (IJCIR). 5(3) (2009a) 297-310.
6. F. Choong, S. Phon-Amnuaisuk, and M.Y. Alias, Implementation of Local Search in Hybrid Multi-Objective Adaptive GA: A Case Study on High-Level Synthesis, International Journal of Computational Intelligence Research. 5(3), (2009b) 309-326.
7. F. Choong, S. Phon-Amnuaisuk, and M.Y. Alias, Metaheuristic Methods in Hybrid Flow Shop Scheduling Problem, Expert Systems with Applications. 38(9) (2011) 10787-10793.
8. H. Ishibuchi, and T. Murata, A multi-objective genetic local search algorithm and its application to flowshop scheduling, IEEE Transactions on Systems, Man, and Cybernetics - Part C: Applications and Reviews. 28(3) (1998) 392-403.
9. Essafi, Y. Matib, and S. Dauzère-Pérès, A genetic local search algorithm for minimizing total weighted tardiness in the job-shop scheduling problem, Computers & Operations Research. 35(8) (2008) 2599-2616.
10. J. Chen and S. Ho, Multi-Objective Evolutionary Optimization of Flexible Manufacturing Systems, Proceedings of the Genetic and Evolutionary Computation Conference, July 7-11, San Francisco, California. (2001) 1260–1267.
11. J. U. Kim, and Y. D. Kim, Simulated annealing and genetic algorithms for scheduling problems with multi-level product structure, Computers and Operations Research. 23(9) (1996) 857–868.
12. L.E. Ulungu, J. Teghem, P.H. Fortemps, and D. Tuyttens, MOSA method: a tool for solving multiobjective combinatorial optimization problems, Journal of Multi-criteria Decision Analysis. 8(5) (1999) 221-236.
13. L. Wang, and D. Z. Zheng, An efective hybrid optimization strategy for job-shop scheduling problems, Computers and Operations Research. 28(3) (2001) 585–596.
14. R. Kumar, and P. Rockett, Improved sampling of the Pareto-front in multi-objective genetic optimization by steady-stage evolution: a Pareto converging genetic algorithm. Evolutionary Computation. (2009) 10(6) 283-314.

15. S.G. Ponnambalam, and S.K. Low, Solving Machine Loading Problem in Flexible Manufacturing Systems Using Particle Swarm Optimization, World Academy of Science, Engineering and Technology. 39(8) (2008) 14-19.
16. X. J. Wei, and Z. M. Wu, An effective hybrid optimization approach for multi-objective flexible jobshop scheduling problems, Computers and Industrial Engineering. 48(1) (2005) 409–425.

Authors

Florence Choong Chiao Mei received the BEng. (First class) from Multimedia University, Malaysia in 2002. She then completed her Masters of Engineering Science (MEngSc) degree in Multimedia University, Cyberjaya in 2005 and PhD in Engineering in 2012. Recently, she has completed her Masters in Business Administration (MBA) from the University of Derby, UK.
She is presently the Head of Programme and a Senior Lecturer at the School of Engineering, Taylor's University, Malaysia. She is author and co-author of numerous international journal and conference papers published by renowned journals in power quality, VLSI system design and artificial intelligence. Her current research interests are in the area of artificial intelligence and VLSI design.

Somnuk Phon-Amnuaisuk received his B.Eng (H) from King Mongkut Institute of Technology and Ph.D. in Artificial Intelligence from the University of Edinburgh. Previously, he was an Associate Dean at the faculty of Information Technology, Multimedia University, Malaysia where he is the chairman for Centre of Artificial Intelligence and Intelligent Computing. Dr Somnuk has also served as a committee member in many editorial boards and research grant screening committees. He is currently attached to the Faculty of Business and Computing, Institut Teknologi Brunei.

Mohamad Yusoff Alias obtained the Bachelor of Science in Engineering (Electrical Engineering) degree from the University of Michigan, Ann Arbor, in May 1998. He then received his Ph.D. degree in December 2004 from the School of ECS, University of Southampton in the United Kingdom. He is currently an Associate Dean in the Faculty of Engineering, Multimedia University in Malaysia. His research interests cover the field of wireless communications especially in OFDM, multiple antenna system, multiuser detection, genetic algorithms in communications, multimedia applications and wireless security.