Analyzing the Tolerance for Imprecise Representation of Processing Times in Job-Shop Scheduling

Thanh-Do Tran

Dolphin Team
Inria Research Center
Lille – Nord Europe
59650 Villeneuve d'Ascq, France
thanh-do.tran@inria.fr

version of February 9, 2014
compiled on May 23, 2014

Abstract

We have experimented with 12 famous job-shop scheduling problem instances. For each of the crisp instance, a total of 1000 fuzzy instances were sampled/generated in such a way that TODO: explain. We then launched the algorithms 30 times on each fuzzy instance. The output of the 30 runs were generalized by a summary statistic, which could be the minimum or the median over the 30 values. The minimum represents the best effort, while the median is considered as a more reliable representative of an algorithm's runs. There algorithms are GA optimizing the expected makespan, NSGA-II optimizing the 3 defining points of the triangular makespan, and a Memetic Algorithm constructed from the NSGA-II and a Local Search using the N2 neighborhood structure. All the algorithms used a population size of 500 running within an equivalent budget of 200 generations. We are interested in investigating the performance of the Memetic Algorithm in comparison with the others over the different problem instances. We also wish to gain some insight into the progress of the algorithms, i.e. the quality of solutions along the runs. Moreover, we want to see whether and how the number of fuzzy instances generated may affect our conclusion regarding the algorithms' superiority.

TODO: Add some statistical test on each table

Table of Contents

To be added

Table 1: Expected makespan (mean and SD) over 1000 fuzzified instances of 5 algorithms at a budget of 250 generations. Each algorithm was run 30 times and the *min* is considered. Numbers in brackets are standard deviation (SD).

Instance	**crisp-GA**	**fuzzy-GA**	**f-NSGA2**	**f-GA-LS**	**f-NSGA2-LS**
FT10	947.2 (7.6)	945.8 (7.4)	942.7 (6.2)	942.4 (6.7)	940.3 (5.8)
FT20	1188.3 (7.0)	1184.6 (6.7)	1179.4 (6.7)	1180.8 (6.5)	1178.5 (6.7)
LA21	1086.1 (7.3)	1082.9 (7.1)	1076.8 (6.4)	1071.5 (6.0)	1067.0 (5.6)
LA24	980.6 (7.2)	977.3 (7.1)	970.8 (6.8)	958.7 (6.3)	954.6 (5.8)
LA25	1010.2 (6.3)	1008.1 (6.1)	1003.2 (5.6)	997.5 (5.4)	994.2 (4.9)
LA27	1306.0 (7.7)	1302.9 (8.2)	1292.5 (7.6)	1279.5 (6.5)	1273.3 (5.5)
LA29	1237.7 (8.2)	1236.9 (8.6)	1226.5 (7.8)	1225.6 (8.7)	1216.3 (7.8)
LA38	1279.1 (10.4)	1275.1 (10.5)	1263.5 (9.6)	1252.8 (10.1)	1244.4 (9.3)
LA40	1279.8 (8.5)	1276.1 (8.6)	1267.5 (7.4)	1256.2 (7.9)	1250.9 (7.4)
ABZ7	715.2 (4.4)	713.4 (4.6)	707.8 (4.1)	696.9 (4.3)	692.6 (4.1)
ABZ8	738.3 (5.0)	736.4 (4.9)	729.3 (4.9)	712.9 (4.4)	707.3 (4.3)
ABZ9	744.4 (5.2)	742.8 (5.1)	735.6 (4.8)	729.7 (4.9)	724.1 (4.8)

Table 2: Expected makespan (mean and SD) over 1000 fuzzified instances of 5 algorithms at a budget of 250 generations. Each algorithm was run 30 times and the *median* is considered. Numbers in brackets are standard deviation (SD).

Instance	**crisp-GA**	**fuzzy-GA**	**f-NSGA2**	**f-GA-LS**	**f-NSGA2-LS**
FT10	977.5 (5.2)	974.9 (5.2)	968.0 (5.2)	970.2 (4.6)	965.3 (4.4)
FT20	1210.8 (5.6)	1208.1 (5.8)	1196.9 (5.7)	1201.6 (5.8)	1193.6 (5.5)
LA21	1115.6 (5.3)	1111.9 (5.4)	1100.9 (4.9)	1093.7 (4.8)	1087.0 (4.6)
LA24	1007.2 (4.7)	1003.7 (4.6)	995.3 (4.6)	982.8 (4.2)	976.5 (4.2)
LA25	1036.0 (5.0)	1033.8 (5.3)	1024.6 (4.8)	1023.4 (5.0)	1015.1 (5.3)
LA27	1339.5 (5.1)	1336.8 (5.4)	1322.6 (5.0)	1305.6 (4.8)	1295.6 (4.6)
LA29	1270.1 (5.3)	1269.3 (5.4)	1255.8 (4.9)	1257.8 (5.2)	1247.6 (5.0)
LA38	1321.8 (6.3)	1317.9 (6.3)	1302.0 (6.1)	1292.8 (6.0)	1281.9 (5.7)
LA40	1317.5 (5.8)	1313.3 (5.9)	1300.2 (5.7)	1284.6 (5.6)	1276.8 (5.5)
ABZ7	733.5 (2.7)	732.0 (2.7)	724.9 (2.6)	713.7 (2.7)	707.6 (2.6)
ABZ8	759.1 (3.0)	757.4 (2.9)	749.5 (2.8)	730.3 (2.6)	723.7 (2.6)
ABZ9	765.9 (2.9)	764.5 (3.0)	755.9 (2.9)	749.6 (2.9)	742.8 (2.8)