Appendix Overview of the instances

Fisher and Thompson

H. Fisher and G. L. Thompson. *Probabilistic Learning Combinations of Local Job-Shop Scheduling Rules.* In: *Industrial Scheduling*, **15**: *225–251.* ed. by J. F. Muth and G. L. Thompson. Prentice Hall, 1963

Instance	# jobs	# machines	Lower bound	Upper bound
ft06	6	6	55[16] [a]	**55**[16] [a]
ft10	10	10	930[8] [b]	**930** [c]
ft20	20	5	1165[28]	**1165**[28]

[8] Carlier and Pinson (1989)
[16] Florian, Trepant, and McMahon (1971)
[28] McMahon and Florian (1975)
[a] Using algorithms of Schrage [37] and Balas [3]
[b] Achieved in 1986 [see 1]
[c] B.J. Lageweg (1984) [see 24]

Table 1: Instances of Fisher and Thompson [15]

Lawrence

S. Lawrence. *Resource Constrained Project Scheduling: An Experimental Investigation of Heuristic Scheduling Techniques (Supplement).* Carnegie-Mellon University, 1984

Instance	# jobs	# machines	Lower bound	Upper bound
la01	10	5	666[1]	**666**[1]
la02	10	5	655[1]	**655**[27]
la03	10	5	597[2]	**597**[27]
la04	10	5	590[2]	**590**[27]
la05	10	5	593[1]	**593**[1]
la06	15	5	926[1]	**926**[1]
la07	15	5	890[1]	**890**[1]
la08	15	5	863[1]	**863**[1]
la09	15	5	951[1]	**951**[1]
la10	15	5	958[1]	**958**[27]

[1] Adams, Balas, and Zawack (1988)
[2] Applegate and Cook (1991)
[27] Matsuo, Suh, and Sullivan (1988)

Table 2: Instances of Lawrence [25]

Instance	# jobs	# machines	Lower bound	Upper bound
la11	20	5	1222[1]	**1222**[1]
la12	20	5	1039[1]	**1039**[1]
la13	20	5	1150[1]	**1150**[1]
la14	20	5	1292[1]	**1292**[1]
la15	20	5	1207[1]	**1207**[1]
la16	10	10	945[9]	**945**[9]
la17	10	10	784[9]	**784**[27]
la18	10	10	848[2]	**848**[27]
la19	10	10	842[2]	**842**[27]
la20	10	10	902[2]	**902**[2]
la21	15	10	1046[45]	**1046**[45]
la22	15	10	927[2]	**927**[27]
la23	15	10	1032[1]	**1032**[1]
la24	15	10	935[2]	**935**[2]
la25	15	10	977[2]	**977**[2]
la26	20	10	1218[1]	**1218**[27]
la27	20	10	1235[1]	**1235**[10]
la28	20	10	1216[1]	**1216**[2]
la29	20	10	1152[26]	**1152**[26]
la30	20	10	1355[1]	**1355**[1]
la31	30	10	1784[1]	**1784**[1]
la32	30	10	1850[1]	**1850**[1]
la33	30	10	1719[1]	**1719**[1]
la34	30	10	1721[1]	**1721**[1]
la35	30	10	1888[1]	**1888**[1]
la36	15	15	1268[9]	**1268**[9]
la37	15	15	1397[2]	**1397**[2]
la38	15	15	1196[45]	**1196**[31]
la39	15	15	1233[2]	**1233**[2]
la40	15	15	1222[2]	**1222**[2]

[1] Adams, Balas, and Zawack (1988)
[2] Applegate and Cook (1991)
[9] Carlier and Pinson (1990)
[10] Carlier and Pinson (1994)
[26] Martin (1996)
[27] Matsuo, Suh, and Sullivan (1988)
[31] Nowicki and Smutnicki (1996)
[45] Vaessens, Aarts, and Lenstra (1996)

Table 2: Instances of Lawrence [25] (continued)

Adams, Balas, and Zawack

Joseph Adams, Egon Balas, and Daniel Zawack. *The Shifting Bottleneck Procedure for Job Shop Scheduling.* Management Science, 34.3: *391–401*, 1988

Instance	# jobs	# machines	Lower bound	Upper bound
abz5	10	10	1234[2]	**1234**[2]
abz6	10	10	943[2]	**943**[1]
abz7	20	15	656[26]	**656**[26]
abz8	20	15	648[46,47]	665[26]
abz9	20	15	678[23]	**678**[50]

[1] Adams, Balas, and Zawack (1988)
[2] Applegate and Cook (1991)
[23] Koshimura et al. (2010)
[26] Martin (1996)
[46] Vilím, Laborie, and Shaw (2015)
[47] Vilím, Laborie, and Shaw (2015)
[50] Zhang et al. (2008)

Table 3: Instances of Adams, Balas, and Zawack [1]

Applegate and Cook

David Applegate and William Cook. *A Computational Study of the Job-Shop Scheduling Problem.* ORSA Journal on Computing, 3.2: *149–156*, 1991

Instance	# jobs	# machines	Lower bound	Upper bound
orb01	10	10	1059[2]	**1059**[2]
orb02	10	10	888[2]	**888**[2]
orb03	10	10	1005[2]	**1005**[2]
orb04	10	10	1005[2]	**1005**[2]
orb05	10	10	887[2]	**887**[2]
orb06	10	10	1010[a]	**1010**[a]
orb07	10	10	397[a]	**397**[a]
orb08	10	10	899[a]	**899**[a]
orb09	10	10	934[a]	**934**[a]
orb10	10	10	944[a]	**944**[a]

[2] Applegate and Cook (1991)
[a] R.J.M. Vaessens using algorithms of [2] (1994) [see 22]

Table 4: Instances of Applegate and Cook [2]

Storer, Wu, and Vaccari

Robert H. Storer, S. David Wu, and Renzo Vaccari. *New Search Spaces for Sequencing Problems with Application to Job Shop Scheduling*. Management Science, 38.10: *1495–1509*, 1992

Instance	# jobs	# machines	Lower bound	Upper bound
swv01	20	10	1407[26]	**1407**[26]
swv02	20	10	1475[26]	**1475**[26]
swv03	20	10	1398[7]	**1398**[44]
swv04	20	10	1464[46,47]	**1464**[46,47]
swv05	20	10	1424[26]	**1424**[26]
swv06	20	15	1630[46,47]	1671[36]
swv07	20	15	1513[46,47]	1594[30]
swv08	20	15	1671[46,47]	1752[11]
swv09	20	15	1633[46,47]	1655[30]
swv10	20	15	1663[46,47]	1743[18]
swv11	50	10	2983[44]	**2983**[33]
swv12	50	10	2972[44]	2977[36]
swv13	50	10	3104[44]	**3104**[43]
swv14	50	10	2968[4]	**2968**[4]
swv15	50	10	2885[44]	**2885**[36]
swv16	50	10	2924[39]	**2924**[39]
swv17	50	10	2794[39]	**2794**[39]
swv18	50	10	2852[39]	**2852**[39]
swv19	50	10	2843[39]	**2843**[39]
swv20	50	10	2823[39]	**2823**[39]

[4] Balas and Vazacopoulos (1994)
[7] Brinkkötter and Brucker (2001)
[11] Cheng, Peng, and Lü (2013)
[18] Gonçalves and Resende (2014)
[26] Martin (1996)
[30] Nagata and Tojo (2009)
[33] Nowicki and Smutnicki (2005)
[36] Peng, Lü, and Cheng (2015)
[39] Storer, Wu, and Vaccari (1992)
[43] Thomsen (1997)
[44] Vaessens (1996)
[46] Vilím, Laborie, and Shaw (2015)
[47] Vilím, Laborie, and Shaw (2015)

Table 5: Instances of Storer, Wu, and Vaccari [39]

Yamada and Nakano

TAKESHI YAMADA and RYOHEI NAKANO. *A genetic algorithm applicable to large-scale job-shop instances.* In: *Parallel instance solving from nature 2*: *281–290*. ed. by REINHARD MÄNNER and BERNARD MANDERICK. Elsevier, 1992.

Instance	# jobs	# machines	Lower bound	Upper bound
yn1	20	20	884[23]	**884**[50]
yn2	20	20	870[7]	904[18]
yn3	20	20	859[46,47]	892[33]
yn4	20	20	929[46,47]	968[43]

[7] Brinkkötter and Brucker (2001)
[18] Gonçalves and Resende (2014)
[23] Koshimura et al. (2010)
[33] Nowicki and Smutnicki (2005)
[43] Thomsen (1997)
[46] Vilím, Laborie, and Shaw (2015)
[47] Vilím, Laborie, and Shaw (2015)
[50] Zhang et al. (2008)

Table 6: Instances of Yamada and Nakano [48]

Taillard

E.D. TAILLARD. *Benchmarks for basic scheduling problems.* European Journal of Operational Research, 64.2: *278–285*, 1993

Instance	# jobs	# machines	Lower bound	Upper bound
ta01	15	15	1231[42,40]	**1231**[42,40]
ta02	15	15	1244[a]	**1244**[31]
ta03	15	15	1218[7]	**1218**[4]
ta04	15	15	1175[7]	**1175**[b]
ta05	15	15	1224[7]	**1224**[7]
ta06	15	15	1238[7]	**1238**[7]
ta07	15	15	1227[7]	**1227**[7]
ta08	15	15	1217[7]	**1217**[4]
ta09	15	15	1274[7]	**1274**[4]
ta10	15	15	1241[a]	**1241**[4]

[4] Balas and Vazacopoulos (1994)
[7] Brinkkötter and Brucker (2001)
[31] Nowicki and Smutnicki (1996)
[40] Taillard (1993)
[42] Taillard (1994)

[a] R.J.M. Vaessens (1995) [see 41]
[b] M. Wennink (1995) [see 41]

Table 7: Instances of Taillard [40]

Instance	# jobs	# machines	Lower bound	Upper bound
ta11	20	15	1357[46,47]	**1357**[34]
ta12	20	15	1367[46,47]	**1367**[4]
ta13	20	15	1342[46,47]	**1342**[19]
ta14	20	15	1345[a]	**1345**[31]
ta15	20	15	1339[46,47]	**1339**[34]
ta16	20	15	1360[46,47]	**1360**[19] [b]
ta17	20	15	1462[c]	**1462**[33] [d]
ta18	20	15	1377[46,47]	1396[4]
ta19	20	15	1332[46,47]	**1332**[34]
ta20	20	15	1348[46,47]	**1348**[34]
ta21	20	20	1642[46,47]	**1642**[6]
ta22	20	20	1561[46,47]	1600[33] [e]
ta23	20	20	1518[46,47]	1557[33] [e]
ta24	20	20	1644[46,47]	**1644**[6]
ta25	20	20	1558[46,47]	1595[33] [d]
ta26	20	20	1591[46,47]	1643[6]
ta27	20	20	1652[46,47]	1680[33] [e]
ta28	20	20	1603[46,47]	**1603**[50]
ta29	20	20	1573[46,47]	1625[f]
ta30	20	20	1519[46,47]	1584[33] [e]
ta31	30	15	1764[40]	**1764**[g]
ta32	30	15	1774[40]	1784[h]
ta33	30	15	1788[46,47]	1791[35]
ta34	30	15	1828[40]	1829[33] [e]
ta35	30	15	2007[a]	**2007**[42,40]
ta36	30	15	1819[a]	**1819**[g]
ta37	30	15	1771[40]	**1771**[36]
ta38	30	15	1673[40]	**1673**[b]
ta39	30	15	1795[a]	**1795**[g]
ta40	30	15	1651[46,47]	1669[18]

[4] Balas and Vazacopoulos (1994)
[6] Beck, Feng, and Watson (2011)
[18] Gonçalves and Resende (2014)
[19] Henning (2002)
[31] Nowicki and Smutnicki (1996)
[33] Nowicki and Smutnicki (2005)
[34] Pardalos and Shylo (2006)
[35] Pardalos, Shylo, and Vazacopoulos (2010)
[36] Peng, Lü, and Cheng (2015)
[40] Taillard (1993)
[42] Taillard (1994)
[46] Vilím, Laborie, and Shaw (2015)
[47] Vilím, Laborie, and Shaw (2015)
[50] Zhang et al. (2008)
[a] R.J.M. Vaessens (1995) [see 41]
[b] A. Henning (2000) [see 41]
[c] R. Schilham (2000) [see 41]
[d] Achieved in 2002 [see 41]
[e] Achieved in 2001 [see 41]
[f] E. Aarts (1996) [see 41]
[g] E. Aarts, H. ten Eikelder, J.K. Lenstra and R. Schilham (1999) [see 41]
[h] In [35] (2010) [see 38]. However 1790 is mentioned. 1785 is found in [18]

Table 7: Instances of Taillard [40] (continued)

Instance	# jobs	# machines	Lower bound	Upper bound
ta41	30	20	1906[46,47]	2005[29]
ta42	30	20	1884[46,47]	1937[18]
ta43	30	20	1809[a]	1846[36]
ta44	30	20	1948[46,47]	1979[29]
ta45	30	20	1997[a]	2000[33] [b]
ta46	30	20	1957[46,47]	2004[18]
ta47	30	20	1807[46,47]	1889[36]
ta48	30	20	1912[a]	1941[c]
ta49	30	20	1931[46,47]	1961[29]
ta50	30	20	1833[46,47]	1923[c]
ta51	50	15	2760[42,40]	**2760**[42,40]
ta52	50	15	2756[42,40]	**2756**[42,40]
ta53	50	15	2717[42,40]	**2717**[42,40]
ta54	50	15	2839[42,40]	**2839**[42,40]
ta55	50	15	2679[40]	**2679**[31]
ta56	50	15	2781[42,40]	**2781**[42,40]
ta57	50	15	2943[42,40]	**2943**[42,40]
ta58	50	15	2885[42,40]	**2885**[42,40]
ta59	50	15	2655[42,40]	**2655**[42,40]
ta60	50	15	2723[42,40]	**2723**[42,40]
ta61	50	20	2868[40]	**2868**[31]
ta62	50	20	2869[a]	**2869**[d]
ta63	50	20	2755[40]	**2755**[31]
ta64	50	20	2702[4]	**2702**[31]
ta65	50	20	2725[40]	**2725**[31]
ta66	50	20	2845[40]	**2845**[31]
ta67	50	20	2825[a]	**2825**[21]
ta68	50	20	2784[4]	**2784**[31]
ta69	50	20	3071[40]	**3071**[31]
ta70	50	20	2995[40]	**2995**[31]

[4] Balas and Vazacopoulos (1994)
[18] Gonçalves and Resende (2014)
[21] Jain (1998)
[29] Nagata and Ono (2013)
[31] Nowicki and Smutnicki (1996)
[33] Nowicki and Smutnicki (2005)
[36] Peng, Lü, and Cheng (2015)
[40] Taillard (1993)
[42] Taillard (1994)
[46] Vilím, Laborie, and Shaw (2015)
[47] Vilím, Laborie, and Shaw (2015)
[a] R.J.M. Vaessens (1995) [see 41]
[b] Achieved in 2001 [see 41]
[c] O. V. Shylo (2013) [see 38]
[d] J. P. Caldeira (2003) [see 41]

Table 7: Instances of Taillard [40] (continued)

Instance	# jobs	# machines	Lower bound	Upper bound
ta71	100	20	5464[42,40]	**5464**[42,40]
ta72	100	20	5181[42,40]	**5181**[42,40]
ta73	100	20	5568[42,40]	**5568**[42,40]
ta74	100	20	5339[42,40]	**5339**[42,40]
ta75	100	20	5392[42,40]	**5392**[42,40]
ta76	100	20	5342[42,40]	**5342**[42,40]
ta77	100	20	5436[42,40]	**5436**[42,40]
ta78	100	20	5394[42,40]	**5394**[42,40]
ta79	100	20	5358[42,40]	**5358**[42,40]
ta80	100	20	5183[40]	**5183**[31]

[31] Nowicki and Smutnicki (1996)
[40] Taillard (1993)
[42] Taillard (1994)

Table 7: Instances of Taillard [40] (continued)

Demirkol, Mehta, and Uzsoy

Ebru Demirkol, Sanjay Mehta, and Reha Uzsoy. *Benchmarks for shop scheduling problems.* European Journal of Operational Research, 109.1: *137–141*, 1998

Instance	# jobs	# machines	Lower bound	Upper bound
dmu01	20	15	2501[7]	2563[19]
dmu02	20	15	2651[7]	2706[19]
dmu03	20	15	2731[7]	**2731**[7]
dmu04	20	15	2601[7]	2669[7]
dmu05	20	15	2749[7]	**2749**[7]
dmu06	20	20	3042[20]	3244[35]
dmu07	20	20	2828[20]	3046[35]
dmu08	20	20	3051[a]	3188[35]
dmu09	20	20	2956[a]	3092[19]
dmu10	20	20	2858[a]	2984[34]

[7] Brinkkötter and Brucker (2001)
[19] Henning (2002)
[20] van Hoorn (2016)
[34] Pardalos and Shylo (2006)
[35] Pardalos, Shylo, and Vazacopoulos (2010)
[a] Gharbi and Labidi (2011) using algorithms described in [17] [see 38]

Table 8: Instances of Demirkol, Mehta, and Uzsoy [14]

Instance	# jobs	# machines	Lower bound	Upper bound
dmu11	30	15	3395[12,13,14]	3430[36]
dmu12	30	15	3481[20]	3495[36]
dmu13	30	15	3681[12,13,14]	**3681**[49]
dmu14	30	15	3394[12,13,14]	**3394**[32]
dmu15	30	15	3343[a]	**3343**[21]
dmu16	30	20	3734[a]	3751[18]
dmu17	30	20	3709[a]	3814[b]
dmu18	30	20	3844[12,13,14]	**3844**[18]
dmu19	30	20	3672[20]	3768[36]
dmu20	30	20	3604[12,13,14]	3710[36]
dmu21	40	15	4380[12,13,14]	**4380**[21]
dmu22	40	15	4725[12,13,14]	**4725**[21]
dmu23	40	15	4668[12,13,14]	**4668**[21]
dmu24	40	15	4648[12,13,14]	**4648**[21]
dmu25	40	15	4164[12,13,14]	**4164**[21]
dmu26	40	20	4647[12,13,14]	**4647**[49]
dmu27	40	20	4848[12,13,14]	**4848**[32]
dmu28	40	20	4692[12,13,14]	**4692**[21]
dmu29	40	20	4691[12,13,14]	**4691**[32]
dmu30	40	20	4732[12,13,14]	**4732**[32]
dmu31	50	15	5640[12,13,14]	**5640**[21]
dmu32	50	15	5927[12,13,14]	**5927**[12,13,14]
dmu33	50	15	5728[12,13,14]	**5728**[12,13,14]
dmu34	50	15	5385[12,13,14]	**5385**[12,13,14]
dmu35	50	15	5635[12,13,14]	**5635**[12,13,14]
dmu36	50	20	5621[12,13,14]	**5621**[21]
dmu37	50	20	5851[12,13,14]	**5851**[32]
dmu38	50	20	5713[12,13,14]	**5713**[21]
dmu39	50	20	5747[12,13,14]	**5747**[21]
dmu40	50	20	5577[12,13,14]	**5577**[21]

[12] Demirkol, Mehta, and Uzsoy (1996)
[13] Demirkol, Mehta, and Uzsoy (1997)
[14] Demirkol, Mehta, and Uzsoy (1998)
[18] Gonçalves and Resende (2014)
[20] van Hoorn (2016)
[21] Jain (1998)
[32] Nowicki and Smutnicki (2001)
[36] Peng, Lü, and Cheng (2015)
[49] Zhang et al. (2007)
[a] Gharbi and Labidi (2011) using algorithms described in [17] [see 38]
[b] O. V. Shylo (2013) [see 38]

Table 8: Instances of Demirkol, Mehta, and Uzsoy [14] (continued)

Instance	# jobs	# machines	Lower bound	Upper bound
dmu41	20	15	3007[a]	3248[36]
dmu42	20	15	3224[20]	3390[36]
dmu43	20	15	3292[a]	3441[b]
dmu44	20	15	3299[20]	3488[18]
dmu45	20	15	3039[20]	3272[b]
dmu46	20	20	3575[a]	4035[b]
dmu47	20	20	3522[a]	3939[18]
dmu48	20	20	3447[a]	3763[b]
dmu49	20	20	3403[a]	3710[36]
dmu50	20	20	3496[a]	3729[36]
dmu51	30	15	3954[20]	4167[36]
dmu52	30	15	4094[20]	4311[36]
dmu53	30	15	4141[a]	4394[36]
dmu54	30	15	4202[a]	4362[b]
dmu55	30	15	4146[20]	4271[36]
dmu56	30	20	4554[a]	4941[36]
dmu57	30	20	4302[a]	4655[b]
dmu58	30	20	4319[a]	4708[36]
dmu59	30	20	4219[20]	4624[36]
dmu60	30	20	4319[a]	4755[36]
dmu61	40	15	4917[a]	5172[b]
dmu62	40	15	5041[20]	5265[b]
dmu63	40	15	5111[a]	5326[36]
dmu64	40	15	5130[12,13,14]	5250[b]
dmu65	40	15	5107[20]	5190[b]
dmu66	40	20	5397[20]	5717[36]
dmu67	40	20	5589[a]	5813[b]
dmu68	40	20	5426[a]	5773[36]
dmu69	40	20	5423[a]	5709[36]
dmu70	40	20	5501[a]	5889[b]

[12] Demirkol, Mehta, and Uzsoy (1996)
[13] Demirkol, Mehta, and Uzsoy (1997)
[14] Demirkol, Mehta, and Uzsoy (1998)
[18] Gonçalves and Resende (2014)
[20] van Hoorn (2016)
[36] Peng, Lü, and Cheng (2015)
[a] Gharbi and Labidi (2011) using algorithms described in [17] [see 38]
[b] O. V. Shylo (2013) [see 38]

Table 8: Instances of Demirkol, Mehta, and Uzsoy [14] (continued)

Instance	# jobs	# machines	Lower bound	Upper bound
dmu71	50	15	6080[a]	6223[36]
dmu72	50	15	6395[a]	6483[36]
dmu73	50	15	6001[a]	6163[36]
dmu74	50	15	6123[a]	6220[b]
dmu75	50	15	6029[a]	6197[36]
dmu76	50	20	6342[a]	6813[36]
dmu77	50	20	6499[a]	6822[36]
dmu78	50	20	6586[a]	6770[36]
dmu79	50	20	6650[a]	6970[36]
dmu80	50	20	6459[a]	6686[36]

[36] Peng, Lü, and Cheng (2015)
[a] Gharbi and Labidi (2011) using algorithms described in [17] [see 38]
[b] O. V. Shylo (2013) [see 38]

Table 8: Instances of Demirkol, Mehta, and Uzsoy [14] (continued)

References

[1] Joseph Adams, Egon Balas, and Daniel Zawack. *The Shifting Bottleneck Procedure for Job Shop Scheduling.* Management Science, 34.3: *391–401*, 1988. doi: 10.1287/mnsc.34.3.391 jstor: 2632051

[2] David Applegate and William Cook. *A Computational Study of the Job-Shop Scheduling Problem.* ORSA Journal on Computing, 3.2: *149–156*, 1991. doi: 10.1287/ijoc.3.2.149

[3] Egon Balas. *Machine Sequencing via Disjunctive Graphs: An Implicit Enumeration Algorithm.* Operations Research, 17.6: *941–957*, 1969. doi: 10.1287/opre.17.6.941 jstor: 168317

[4] Egon Balas and Alkis Vazacopoulos. *Guided Local Search with Shifting Bottleneck for Job Shop Scheduling.* Tech. rep. Management Science Research Report, 609. Carnegie Mellon University, 1994

[5] J. E. Beasley. *OR-Library (jobshop1.txt).* url: http://people.brunel.ac.uk/~mastjjb/jeb/orlib/files/jobshop1.txt

[6] J. Christopher Beck, T. K. Feng, and Jean-Paul Watson. *Combining Constraint Programming and Local Search for Job-Shop Scheduling.* INFORMS Journal on Computing, 23.1: *1–14*, 2011. doi: 10.1287/ijoc.1100.0388

[7] Wolfgang Brinkkötter and Peter Brucker. *Solving open benchmark instances for the job-shop problem by parallel head-tail adjustments.* Journal of Scheduling, 4.1: *53–64*, 2001. doi: 10.1002/1099-1425(200101/02)4:1<53::AID-JOS59>3.0.CO;2-Y

[8] J. Carlier and E. Pinson. *An Algorithm for Solving the Job-shop Problem.* Management Science, 35.2: *164–176*, 1989. doi: 10.1287/mnsc.35.2.164 jstor: 2631909

[9] J. Carlier and E. Pinson. *A Practical Use of Jackson's Preemptive Schedule for Solving the Job Shop Problem.* Annals of Operations Research, 26: *269–287*, 1990

[10] J. Carlier and E. Pinson. *Adjustment of heads and tails for the job-shop problem.* European Journal of Operational Research, 78.2: *146–161*, 1994. doi: 10.1016/0377-2217(94)90379-4

[11] T.C.E. Cheng, Bo Peng, and Zhipeng Lü. *A hybrid evolutionary algorithm to solve the job shop scheduling problem.* Annals of Operations Research: *1–15*, 2013. doi: 10.1007/s10479-013-1332-5

[12] Ebru Demirkol, Sanjay V. Mehta, and Reha Uzsoy. *Benchmarking for Shop Scheduling Problems.* Tech. rep. Research memorandum, 96-4. Purdue University, 1996

[13] Ebru Demirkol, Sanjay Mehta, and Reha Uzsoy. *A Computational Study of Shifting Bottleneck Procedures for Shop Scheduling Problems.* Journal of Heuristics, 3.2: *111–137*, 1997. doi: 10.1023/A:1009627429878

[14] Ebru Demirkol, Sanjay Mehta, and Reha Uzsoy. *Benchmarks for shop scheduling problems.* European Journal of Operational Research, 109.1: *137–141*, 1998. doi: 10.1016/S0377-2217(97)00019-2

[15] H. Fisher and G. L. Thompson. *Probabilistic Learning Combinations of Local Job-Shop Scheduling Rules.* In: *Industrial Scheduling*, 15: *225–251*. ed. by J. F. Muth and G. L. Thompson. Prentice Hall, 1963. oclc: 781815542

[16] M. Florian, P. Trepant, and G. McMahon. *An Implicit Enumeration Algorithm for the Machine Sequencing Problem.* Management Science, 17.12: *B-782–B-792*, 1971. doi: 10.1287/mnsc.17.12.B782 jstor: 2629469

[17] Anis Gharbi and Mohamed Labidi. *Extending the Single Machine-Based Relaxation Scheme for the Job Shop Scheduling Problem.* Electronic Notes in Discrete Mathematics, 36: *1057–1064*, 2010. doi: 10.1016/j.endm.2010.05.134

[18] José Fernando Gonçalves and Mauricio G. C. Resende. *An extended Akers graphical method with a biased random-key genetic algorithm for job-shop scheduling.* International Transactions in Operational Research, 21.2: *215–246*, 2014. doi: 10.1111/itor.12044

[19] André Henning. *Praktische Job-Shop Scheduling-Probleme.* PhD thesis. Friedrich-Schiller-Universität Jena, 2002. url: http://www.db-thueringen.de/servlets/DocumentServlet?id=873

[20] Jelke J. van Hoorn. *Dynamic Programming for Routing and Scheduling: Optimizing Sequences of Decisions.* PhD thesis. VU University Amsterdam, 2016. isbn: 978-94-6332-008-5. url: http://dare.ubvu.vu.nl/handle/1871/54396

[21] Anant Singh Jain. *A Multi-Level Hybrid Framework for the Deterministic Job-Shop Scheduling Problem.* PhD thesis. University Of Dundee, 1998. URL: http://www.personal.dundee.ac.uk/~asjain/papers/publications.html

[22] A.S. Jain and S. Meeran. *Deterministic job-shop scheduling: Past, present and future.* European Journal of Operational Research, 113.2: *390–434*, 1999. DOI: 10.1016/S0377-2217(98)00113-1

[23] Miyuki Koshimura, Hidetomo Nabeshima, Hiroshi Fujita, and Ryuzo Hasegawa. *Solving Open Job-Shop Scheduling Problems by SAT Encoding.* IEICE Transactions on Information and Systems, E93.D.8: *2316–2318*, 2010. DOI: 10.1587/transinf.E93.D.2316

[24] Peter J. M. van Laarhoven, Emile H. L. Aarts, and Jan Karel Lenstra. *Job shop scheduling by simulated annealing.* Operations Research, 40.1: *113–125*, 1992. DOI: 10.1287/opre.40.1.113 JSTOR: 171189

[25] S. Lawrence. *Resource Constrained Project Scheduling: An Experimental Investigation of Heuristic Scheduling Techniques (Supplement).* Carnegie-Mellon University, 1984

[26] Paul Douglas Martin. *A time-oriented approach to computing optimal schedules for the job-shop scheduling problem.* PhD thesis. Cornell University, 1996. OCLC: 64683112

[27] Hirofumi Matsuo, Chang Juck Suh, and Robert S. Sullivan. *A Controlled Search Simulated Annealing Method for the General Job-Shop Scheduling Problem.* Working paper 03-04-88. The University of Texas at Austin, 1988

[28] Graham McMahon and Michael Florian. *On Scheduling with Ready Times and Due Dates to Minimize Maximum Lateness.* Operations Research, 23.3: *475–482*, 1975. DOI: 10.1287/opre.23.3.475 JSTOR: 169697

[29] Yuichi Nagata and Isao Ono. *Guided Constructive Local Search for the Job Shop Scheduling Problem.* 2013. submitted

[30] Yuichi Nagata and Satoshi Tojo. *Guided Ejection Search for the Job Shop Scheduling Problem.* In: *Evolutionary Computation in Combinatorial Optimization*: *168–179*. ed. by Carlos Cotta and Peter Cowling. LNCS, 5482. Springer, 2009. ISBN: 978-3-642-01008-8 DOI: 10.1007/978-3-642-01009-5_15

[31] Eugeniusz Nowicki and Czeslaw Smutnicki. *A Fast Taboo Search Algorithm for the Job Shop Problem.* Management Science, 42.6: *797–813*, 1996. DOI: 10.1287/mnsc.42.6.797 JSTOR: 2634595

[32] Eugeniusz Nowicki and Czeslaw Smutnicki. *Some new tools to solve the job-shop problem.* Tech. rep. 60/02. Wroclaw University of Technology, 2001

[33] Eugeniusz Nowicki and Czeslaw Smutnicki. *An Advanced Tabu Search Algorithm for the Job Shop Problem*. Journal of Scheduling, 8.2: *145–159*, 2005. DOI: 10.1007/s10951-005-6364-5

[34] Panos M. Pardalos and Oleg V. Shylo. *An Algorithm for the Job Shop Scheduling Problem based on Global Equilibrium Search Techniques*. Computational Management Science, 3.4: *331–348*, 2006. DOI: 10.1007/s10287-006-0023-y

[35] Panos M. Pardalos, Oleg V. Shylo, and Alkis Vazacopoulos. *Solving job shop scheduling problems utilizing the properties of backbone and "big valley"*. Computational Optimization and Applications, 47.1: *61–76*, 2010. DOI: 10.1007/s10589-008-9206-5

[36] Bo Peng, Zhipeng Lü, and T. C. E. Cheng. *A tabu search/path relinking algorithm to solve the job shop scheduling problem*. Computers & Operations Research, 53: *154–164*, 2015. DOI: 10.1016/j.cor.2014.08.006

[37] Linus Schrage. *Solving Resource-Constrained Network Problems by Implicit Enumeration-Nonpreemptive Case*. Operations Research, 18.2: *263–278*, 1970. DOI: 10.1287/opre.18.2.263 JSTOR: 168683

[38] Oleg V. Shylo. *Job Shop Scheduling at Oleg V. Shylo: Personal Webpage*. URL: http://optimizer.com/jobshop.php

[39] Robert H. Storer, S. David Wu, and Renzo Vaccari. *New Search Spaces for Sequencing Problems with Application to Job Shop Scheduling*. Management Science, 38.10: *1495–1509*, 1992. DOI: 10.1287/mnsc.38.10.1495

[40] E.D. Taillard. *Benchmarks for basic scheduling problems*. European Journal of Operational Research, 64.2: *278–285*, 1993. DOI: 10.1016/0377-2217(93)90182-M

[41] Éric D. Taillard. *Éric Taillard's page*. URL: http://mistic.heig-vd.ch/taillard/problemes.dir/ordonnancement.dir/ordonnancement.html

[42] Éric D. Taillard. *Parallel Taboo Search Techniques for the Job Shop Scheduling Problem*. ORSA Journal on Computing, 6.2: *108–117*, 1994. DOI: 10.1287/ijoc.6.2.108

[43] Søren Thomsen. *Metaheuristikker kombineret med Branch & Bound*. MA thesis. Copenhagen Business School, 1997. OCLC: 464628711

[44] R. J. M. Vaessens. *Addition to the OR-Library [5]*. 1996

[45] R. J. M. Vaessens, E.H.L. Aarts, and J.K. Lenstra. *Job Shop Scheduling by Local Search*. INFORMS Journal on Computing, 8.3: *302–317*, 1996. DOI: 10.1287/ijoc.8.3.302

[46] Petr Vilím, Philippe Laborie, and Paul Shaw. *Failure-Directed Search for Constraint-Based Scheduling*. In: *Integration of AI and OR Techniques in Constraint Programming*: *437–453*. ed. by Laurent Michel. LNCS, 9075. Springer, 2015. ISBN: 978-3-319-18007-6 DOI: 10.1007/978-3-319-18008-3_30

[47] Petr Vilím, Philippe Laborie, and Paul Shaw. *Failure-Directed Search for Constraint-Based Scheduling — Detailed Experimental Results*. 2015. URL: http://vilim.eu/petr/cpaior2015-results.pdf

[48] Takeshi Yamada and Ryohei Nakano. *A genetic algorithm applicable to large-scale job-shop instances*. In: *Parallel instance solving from nature 2*: *281–290*. ed. by Reinhard Männer and Bernard Manderick. Elsevier, 1992. ISBN: 978-0-444-89730-5

[49] ChaoYong Zhang, PeiGen Li, ZaiLin Guan, and YunQing Rao. *A tabu search algorithm with a new neighborhood structure for the job shop scheduling problem*. Computers & Operations Research, 34.11: *3229–3242*, 2007. DOI: 10.1016/j.cor.2005.12.002

[50] ChaoYong Zhang, PeiGen Li, YunQing Rao, and ZaiLin Guan. *A very fast TS/SA algorithm for the job shop scheduling problem*. Computers & Operations Research, 35.1: *282–294*, 2008. DOI: 10.1016/j.cor.2006.02.024