LAPORAN TUGAS AKHIR KOM320 SISTEM CERDAS

OPTIMASI PERMASALAHAN *JOB SHOP SCHEDULING* MENGGUNAKAN *GENETIC ALGORITHM* DAN *CONSTRAINT SATISFACTION PROBLEM*

Disusun oleh:

1. Zahra Aulia Firdausi (G64180030)
2. Putri Melanita Londong Bua (G64180053)
3. Syukriyatul Hanifa (G64180062)
4. Annisa Faradila (G64180074)
5. Hendrika Anggriawan (G64180088)

DEPARTEMEN ILMU KOMPUTER
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
INSTITUT PERTANIAN BOGOR
2021

PENDAHULUAN

A. Latar Belakang

Penjadwalan merupakan proses yang menentukan bagaimana sumber daya dialokasikan ke dalam berbagai operasi yang mungkin. Penjadwalan dapat juga diartikan sebagai suatu proses pengaturan sumber daya untuk menyelesaikan tugas-tugas dengan melibatkan pekerjaan, sumber daya, dan waktu. Dalam hal ini, penjadwalan menjadi suatu hal yang penting untuk sebuah sistem manufaktur. Sebuah jadwal yang baik harus dibentuk untuk mencapai pemanfaatan sistem yang tinggi dan mengurangi biaya produksi dari sistem manufaktur. Dimana suatu produksi membutuhkan berbagai operasi tergantung permintaan dan setiap operasi hanya bisa diselesaikan oleh mesin tertentu. Tujuan dari masalah penjadwalan antara lain, meminimalkan waktu penyelesaian semua tugas (*makespan*), meminimalkan keterlambatan pengerjaan, meminimumkan waktu tunggu pada mesin, meminimumkan biaya, dan lain-lain. Oleh karena itu, diperlukan solusi atau cara yang tepat agar jadwal yang dihasilkan efisien, memenuhi aspek keadilan, dan tidak memerlukan waktu pemrosesan yang panjang. Salah satu model matematika yang dapat digunakan untuk menyelesaikan permasalahan penyusunan jadwal adalah *job shop scheduling*. Masalah penjadwalan *job shop* merupakan salah satu masalah penjadwalan yang memiliki kendala urutan pemrosesan operasi. Masalah penjadwalan *job shop* adalah penjadwalan yang melibatkan suatu tugas pada seperangkat kerja pada stasiun-kerja (mesin) secara sekuensial, saat mengoptimalkan satu atau lebih sasaran tanpa melanggar batasan yang diterapkan pada *job shop* [2].

Masalah penjadwalan *job shop* merupakan persoalan mengurutkan sejumlah operasi yang diproses pada mesin-mesin tertentu [1]. Masalah penjadwalan *job shop* adalah bagaimana menyusun semua operasi dari semua *job* pada tiap mesin dalam rangka meminimasi fungsi objektif. Fungsi obyektif yang dimaksud dapat berupa waktu pengerjaan total, rata-rata waktu pengerjaan, rata-rata waktu keterlambatan penyelesaian *job*, atau lainnya [3]. Berdasarkan waktu kedatangan *job*, penjadwalan *job shop* dapat dikelompokkan sebagai *static job shop scheduling* dan *dynamic job shop scheduling*. Pada *static job shop scheduling*, semua *job* diterima pada saat yang sama. Pada *dynamic job shop scheduling*, waktu kedatangan *job* bervariasi tetapi sudah diketahui sebelumnya (*deterministic*) atau waktu kedatangan *job* bervariasi dan tidak dapat diketahui sebelumnya (*non deterministic*/*stochastic*) [4].

Terdapat beberapa *constraint* atau batasan-batasan tertentu yang harus dipenuhi agar mampu memberikan solusi yang paling optimal. Diperlukan optimasi yang sesuai mengingat banyaknya ketentuan atau syarat yang harus dipenuhi dilakukan guna menghindari terjadinya *overlap*. Oleh karena itu, pada penelitian ini kami mencoba melakukan optimasi masalah *job shop scheduling* tersebut dengan menggunakan *genetic algorithm* dan *constraint satisfaction problem*.

B. Tujuan

Penelitian ini bertujuan untuk menemukan solusi yang paling optimal dalam permasalah *job shop scheduling* dengan membandingkan *genetic algorithm* dan *backtracking algorithm* yang didapat dari penelitian sebelumnya.

C. Ruang Lingkup

Ruang lingkup pada penelitian ini adalah sebagai berikut:

1. Data yang digunakan berasal dari situs http://mistic.heig-vd.ch/. Data yang dipakai adalah jenis mesin, lama waktu penggunaan mesin, dan jumlah pekerja.
2. Pendekatan yang digunakan untuk optimasi permasalahan adalah *genetic algorithm* dan *constraint satisfaction problem* (CSP).
3. Proses optimasi menggunakan bahasa pemrograman Python.

METODE PENELITIAN

Perancangan sistem pada penelitian ini dilakukan dalam empat tahap, yaitu tahap pengumpulan data, tahap pendefinisian constraint, tahap penyusunan pendekatan, dan tahap evaluasi.

1. Pengumpulan Data

   Data yang digunakan pada penelitian ini bersumber dari situs http://mistic.heig-vd.ch/. Data yang dipakai adalah jenis mesin, lama waktu penggunaan mesin, dan jumlah pekerja. Terdapat 15 pekerja yang akan menggunakan 15 mesin dengan durasi pemakaian yang berbeda.

2. Pendefinisian *Constraint*

   Berikut beberapa *constraint* yang harus dipenuhi dalam menyusun *job shop scheduling* yaitu:
   - Suatu *task* tidak akan dimulai jika *task* sebelumnya dari pekerja yang sama belum selesai dilakukan.
   - Sebuah mesin hanya dapat melakukan satu *task* pada waktu yang sama.
   - Jika suatu *task* sudah dijalankan, maka *task* tersebut harus dijalankan hingga selesai.

3. Penyusunan Pendekatan

   Pendekatan yang digunakan dalam mengoptimasi permasalahan *job shop scheduling* adalah *genetic algorithm* dan *constraint satisfaction problem* (CSP). *Genetic algorithm* merupakan salah satu algoritma yang sering digunakan untuk menyelesaikan permasalahan CSP.

   Genetic algorithm adalah algoritma yang menerapkan pemahaman evolusi alam untuk *task* pemecahan masalah [6]. Pendekatan yang dilakukan oleh algoritma ini adalah dengan menggabungkan secara acak berbagai pilihan solusi optimal dalam suatu kumpulan untuk mendapatkan generasi solusi terbaik berikutnya, yaitu dalam kondisi yang memaksimalkan kesesuaiannya dan sesuai dengan *constraint* yang telah ditentukan [5].

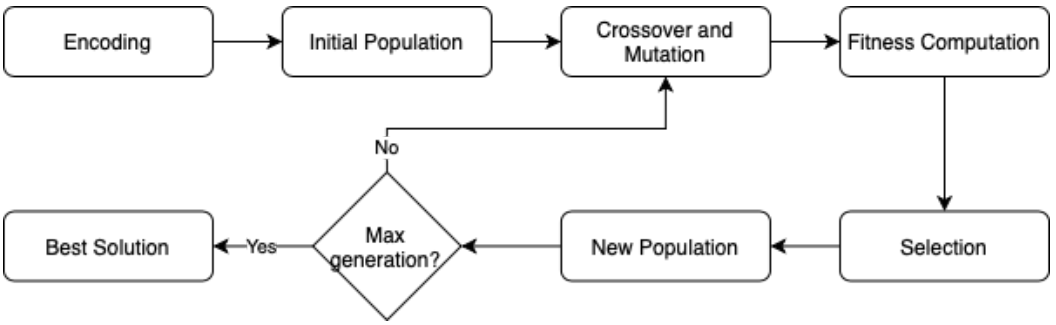
4. Evaluasi

   Pada tahap evaluasi dilakukan dengan membandingkan lama waktu penyelesaian menggunakan *backtracking algorithm* dengan penyelesaian menggunakan *genetic algorithm* untuk mengetahui kesesuaian penggunaan CSP dengan *genetic algorithm* pada kasus *job shop scheduling*.

HASIL DAN PEMBAHASAN

A. Perancangan Sistem

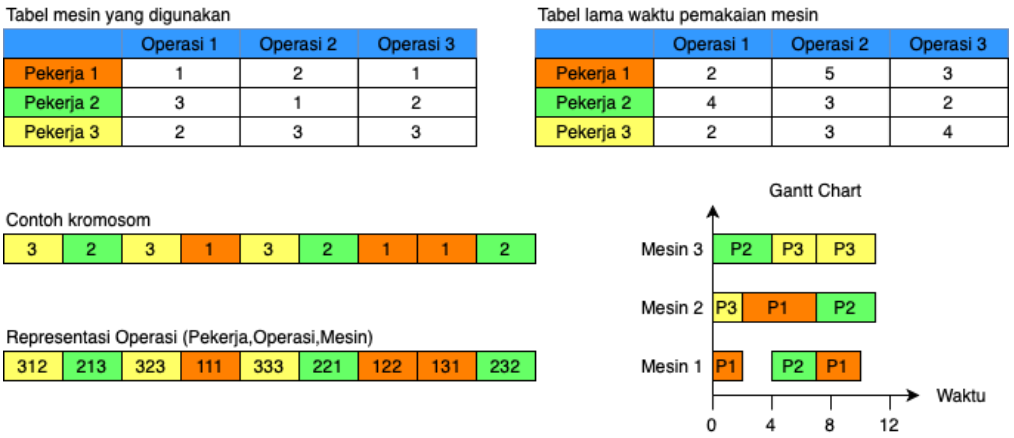
Program *genetic algorithm* yang kami susun memiliki cara kerja sebagai berikut:



Gambar 1. Flowchart Proses Algoritma *Job Shop Scheduling*

1. *Encoding*

Pengkodean gen dalam kromosom dilakukan dengan menggunakan metode Gen-Tsujimura-Kubota (1994, 1997). Dalam program ini, kromosom akan bertindak sebagai urutan operasi yang dilakukan sedangkan gennya akan bertindak sebagai operasi yang dilakukan. Berikut merupakan penjelasan proses *encoding* dengan menggunakan contoh:



Gambar 2. Ilustrasi *Encoding*

Pada gambar di atas terdapat dua tabel. Tabel kiri merupakan tabel penggunaan mesin, sedangkan tabel kanan merupakan tabel lama waktu pemakaian mesin. Pada contoh tersebut terdapat 3 pekerja dan 3 mesin. Setiap pekerja harus melakukan 3 operasi yang dilakukan secara berurutan, contoh pekerja 1 melakukan operasi 1 dengan menggunakan mesin 1 selama 2 satuan waktu, lalu melakukan operasi 2 dengan menggunakan mesin 2 selama 5 satuan waktu, lalu melakukan operasi 3 dengan menggunakan mesin 1 selama 3 satuan waktu.

Kromosom menunjukkan urutan operasi yang akan dilakukan. Kromosom berisi kumpulan gen yang menunjukkan operasinya. Pada contoh di atas urutan gennya adalah 3,2,3,1,3,2,1,1,2. Artinya pekerja 3 akan melakukan operasi 1, lalu pekerja 2 akan melakukan operasi 1, lalu pekerja 3 akan melakukan operasi 2, lalu

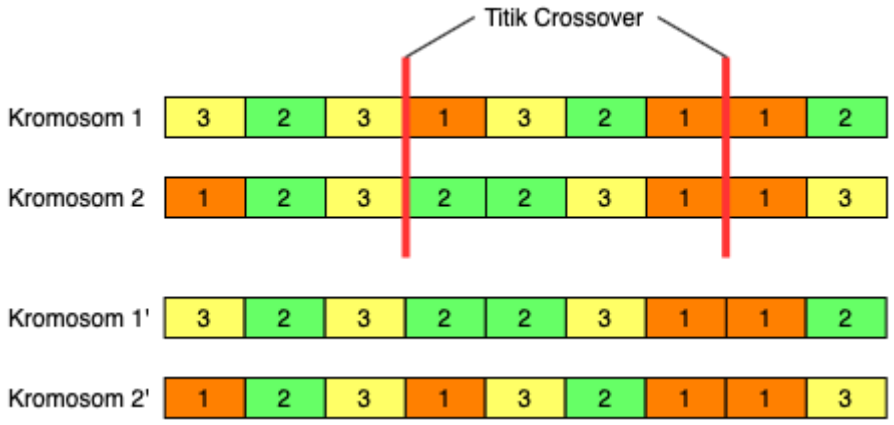
pekerja 1 akan melakukan operasi 1, dan seterusnya. Representasi urutannya bisa dilihat pada *gantt chart* di sebelah kanan bawah.

2. *Initial Population*

   Pembangkitan populasi awal berupa kumpulan kromosom sepanjang banyaknya tugas yang diinputkan. Dalam program ini, populasi awal yang dihasilkan akan dianggap sebagai solusi pertama dari *genetic algorithm* yang berikutnya akan diperbaiki oleh proses-proses berikutnya. Pembangkitan populasi awal ini dilakukan dengan metode *random*.

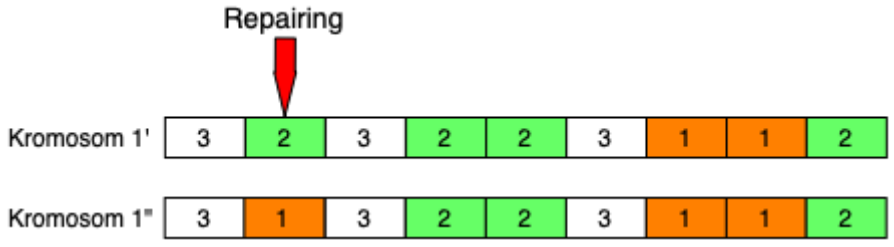
3. *Crossover* dan *Mutation*

   Crossover merupakan proses persilangan 2 kromosom *parent* yang menghasilkan suatu kromosom baru. *Crossover* yang dilakukan pada program merupakan *crossover* 2 titik.



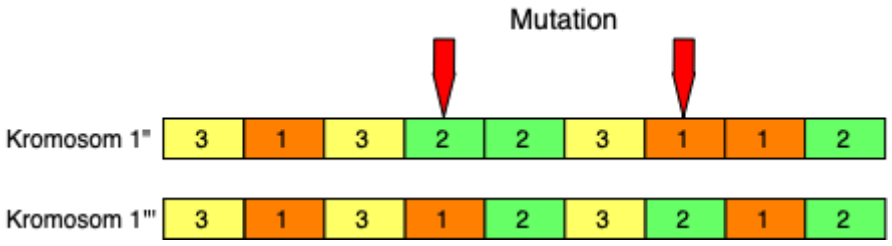
Gambar 3. Ilustrasi *Crossover* 2 Titik

Setelah proses *crossover*, dilakukan proses *repairing*. *Repairing* diperlukan karena setiap kromosom yang ada harus memenuhi syarat bahwa jumlah kemunculan gen yang sama harus sama dengan banyaknya operasi yang dilakukan oleh pekerja yang bersesuaian. Pada contoh ini, setiap angka hanya boleh muncul 3 kali dalam satu kromosom.



Gambar 4. Ilustrasi *Repairing* pada Kromosom

Mutation merupakan proses mengubah beberapa gen di dalam kromosom baru hasil *crossover*. Di dalam algoritma ini, proses *mutation* dilakukan dengan menukar posisi gen dalam kromosom yang sama agar kromosom tetap memenuhi syarat yang ada.



Gambar 5. Ilustrasi *Mutation*

4. *Fitness Computation*

   Semua kromosom hasil dari proses sebelumnya akan dihitung nilai *fitness*-nya. Dalam hal ini, nilai *fitness* yang dimaksud menunjukkan waktu yang dibutuhkan untuk menyelesaikan seluruh tugas yang ada. Semakin kecil waktu yang dibutuhkan maka nilai *fitness*-nya semakin baik.

5. *Selection*

   Tidak semua kromosom akan lanjut ke proses berikutnya. Kromosom-kromosom diseleksi dengan metode *Roulette Wheel*.

6. *New Population*

   Setelah proses 3 sampai 5, didapatkan populasi baru. Ada dua hal yang akan dilakukan pada populasi yang baru ini. Yang pertama, selama batas generasi belum tercapai maka populasi ini akan mengulang proses 3 sampai 5. Yang kedua, jika batas generasi telah tercapai, maka proses iterasi akan berhenti dan lanjut ke proses berikutnya.

7. *Best Solution*

   Setelah mencapai batas generasi, didapatkan populasi dengan nilai *fitness* terbaik. Dari populasi ini, hanya akan dipilih satu kromosom dengan nilai *fitness* terbaik saja yang akan menjadi output dari keseluruhan algoritma.

B. Uji Coba Sistem

Kriteria Penjadwalan

Pada proses uji coba terdapat kriteria penjadwalan yaitu sebagai berikut:

1. Input penjadwalan pekerja berupa alokasi mesin yang bekerja secara *dependent* dan durasi pemakaiannya.
2. Jadwal penggunaan mesin dan pekerja yang dibutuhkan untuk menyelesaikan sebuah *task* akan diatur sedemikian rupa agar tidak terjadi tabrakan antara penggunaan mesin yang satu dengan yang lainnya.
3. Output penjadwalan berupa jadwal yang optimal.

Penjadwalan Produksi

Penjadwalan produksi yang dilakukan menggunakan 15 jenis mesin dengan alokasi waktu yang berbeda. Kemudian terdapat 15 pekerja yang harus menyelesaikan semua *task* dengan jumlah mesin yang sudah ditentukan.

Pada tahap ini, hal yang dilakukan adalah pengujian. Berikut adalah lama waktu (detik) penggunaan mesin setiap pekerja.

		Pekerja 0	Pekerja 1	Pekerja 2	Pekerja 3	Pekerja 4	Pekerja 5	Pekerja 6	Pekerja 7	Pekerja 8	Pekerja 9	Pekerja 10	Pekerja 11	Pekerja 12	Pekerja 13	Pekerja 14
Task 1	Mesin	7	5	2	6	8	6	13	12	11	7	5	3	6	9	11
	Waktu	94	74	4	73	78	29	18	32	85	5	90	47	65	28	57
Task 2	Mesin	13	6	9	3	9	4	4	6	12	12	8	15	9	15	9
	Waktu	66	31	82	23	23	61	75	52	30	59	27	43	62	21	16
Task 3	Mesin	5	8	10	10	7	13	8	1	7	10	14	1	11	5	13
	Waktu	10	88	40	30	21	88	20	9	96	30	1	75	97	51	42
Task 4	Mesin	8	15	13	7	11	14	9	8	15	3	1	13	3	14	7
	Waktu	53	51	86	30	60	70	4	49	91	60	8	8	20	75	34
Task 5	Mesin	4	14	7	11	5	12	15	13	1	9	6	7	4	6	5
	Waktu	26	57	50	53	36	16	91	61	13	41	91	51	31	17	37
Task 6	Mesin	3	9	12	1	10	5	7	14	2	1	13	11	7	7	2
	Waktu	15	78	54	94	29	31	68	35	87	17	80	3	33	89	26
Task 7	Mesin	11	12	14	14	3	15	2	15	3	14	7	8	10	10	14
	Waktu	65	8	21	58	95	65	19	99	82	66	89	84	33	59	68
Task 8	Mesin	12	10	6	5	15	8	12	2	6	4	9	6	1	2	15
	Waktu	82	7	6	93	99	83	54	62	83	89	49	34	77	56	73
Task 9	Mesin	9	7	1	8	13	3	5	3	13	11	15	9	14	13	12
	Waktu	10	91	54	32	79	78	85	6	78	78	32	28	50	63	5
Task 10	Mesin	15	11	3	15	6	2	6	9	5	8	11	10	5	8	1
	Waktu	27	79	68	91	76	26	73	62	56	88	28	60	80	18	8
Task 11	Mesin	10	1	8	12	2	11	3	5	9	2	4	14	2	12	8
	Waktu	93	18	82	30	93	50	43	7	85	69	90	69	48	17	12
Task 12	Mesin	14	4	11	9	14	1	11	4	8	13	2	2	12	11	4
	Waktu	92	51	20	56	42	87	24	80	8	45	93	45	90	30	87
Task 13	Mesin	6	13	5	13	12	10	1	10	10	15	12	4	13	4	3
	Waktu	96	18	39	27	52	62	37	3	66	82	6	67	75	16	83
Task 14	Mesin	1	2	4	2	1	7	14	7	14	5	10	12	8	3	10
	Waktu	70	99	35	92	42	14	87	57	88	6	35	58	96	7	20
Task 15	Mesin	2	3	15	4	4	9	10	11	4	6	3	5	15	1	6
	Waktu	83	33	68	9	96	30	66	7	15	13	73	87	44	35	97

Gambar 6. Lama Waktu Penggunaan Mesin Setiap Pekerja dalam Detik

Pengolahan Data dengan *Genetic Algorithm*

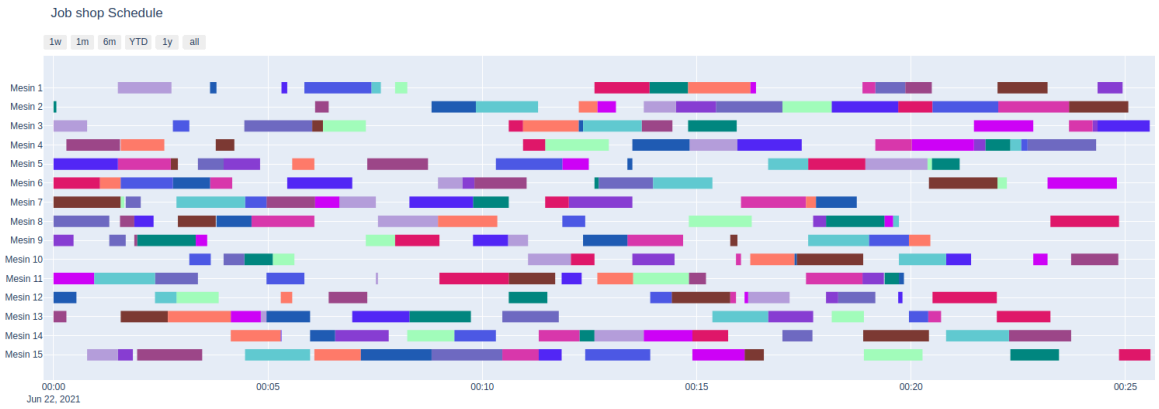
Setelah data di atas dimasukkan ke dalam program didapatkan keluaran berupa penjadwalan untuk menyelesaikan semua *task* dengan waktu paling optimal. Untuk mendapatkan *output* yang paling optimal, kami mencoba mengubah nilai parameternya yang terdiri dari nilai populasi, *crossover*, mutasi, seleksi dan banyaknya generasi.

Tes percobaan pada penelitian ini akan diuji sebanyak 16 kali untuk mendapatkan hasil kromosom dengan nilai *makespan* yang terbaik untuk kasus dengan 15 pekerja dan 15 mesin. Untuk mendapatkan nilai *makespan* yang minimum, kami akan melakukan percobaan pada parameter mutasi dan *crossover*. Nilai *crossover* yang akan dipakai yaitu 0.2, 0.4, 0.6, dan 0.8, sedangkan untuk mutasi nilai yang akan dipakai yaitu 0.02, 0.04, 0.06, dan 0.08. Berikut tabel hasil percobaan:

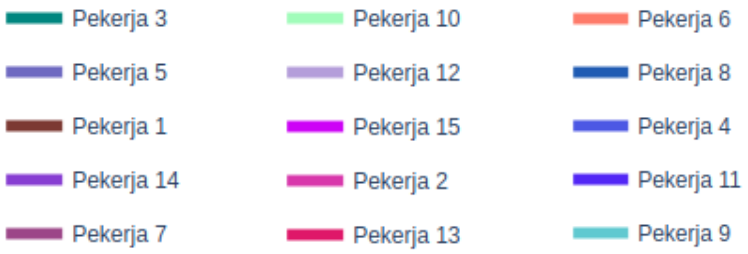
Tabel 1. Tabel Perbandingan Nilai *Crossover* dan Mutasi Terhadap Nilai *Makespan*

No	*Crossover*	Mutasi	Populasi	Seleksi	Generasi	*Makespan* (detik)
1	0.2	0.02	100	0.1	1000	1709
2		0.04	100	0.1	1000	1706
3		0.06	100	0.1	1000	1760
4		0.08	100	0.1	1000	1636
5	0.4	0.02	100	0.1	1000	1720
6		0.04	100	0.1	1000	1730
7		0.06	100	0.1	1000	1754
8		0.08	100	0.1	1000	1653
9	0.6	0.02	100	0.1	1000	1729
10		0.04	100	0.1	1000	1761
11		0.06	100	0.1	1000	1659
12		0.08	100	0.1	1000	1727
13	0.8	0.02	100	0.1	1000	1535
14		0.04	100	0.1	1000	1644
15		0.06	100	0.1	1000	1723
16		0.08	100	0.1	1000	1784

Dapat dilihat dari tabel diatas, nilai *makespan* yang paling minimum yaitu bernilai 1535 (detik) dengan nilai parameter populasi sebesar 100, nilai *crossover* sebesar 0.8, nilai mutasi sebesar 0.02, nilai seleksi sebesar 0.1 dengan jumlah generasi sebanyak 1000. Kemudian, akan divisualisasikan pada grafik seperti pada *gantt chart* di bawah ini:

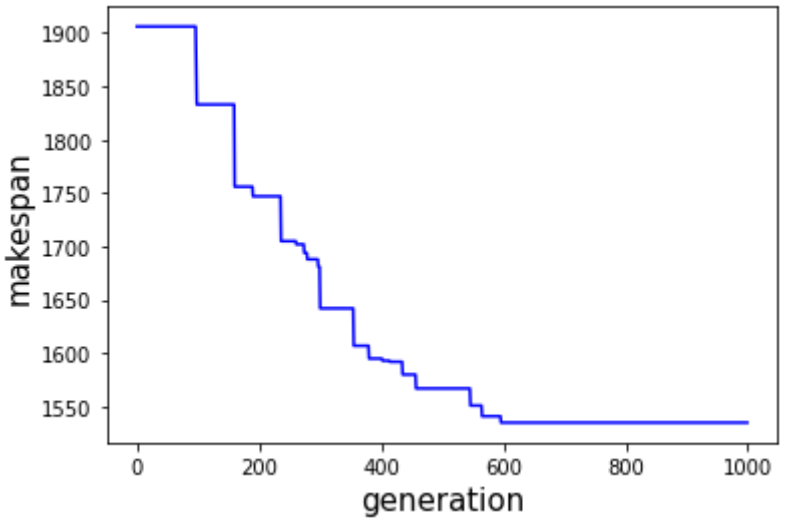


Gambar 7. *Gantt Chart Job Shop Scheduling* dengan *Genetic Algorithm*



Gambar 8. Daftar Indeks Warna Pekerja pada *Job Shop Scheduling*

Dari hasil penjadwalan yang didapat pada Gambar 7, pekerjaan dimulai pada waktu 00.00 dan berakhir pada jam 00.25 yang artinya semua pekerjaan dapat diselesaikan dalam waktu 1535 detik atau 25 menit 35 detik. Setiap mesin akan memiliki urutan pekerja yang berbeda-beda yang artinya pemakaian mesin tidak akan mengalami tabrakan. Misalkan pada mesin 1 urutan pekerjanya yaitu 12-8-11-4-9-10-13-3-6-15-2-5-7-1-14, sedangkan untuk mesin 2 urutan pekerja yaitu 3-7-8-9-6-15-12-14-5-10-11-13-4-2-1, dan seterusnya sampai dengan mesin 15.



Gambar 9. Grafik *Makespan*

Dari keadaan ini dapat dianalisis bahwa *best makespan* yang dihasilkan yaitu sebesar 1535 detik yang berada pada rentang 600 sampai tak hingga generasi. Sedangkan *worst makespan* yang dihasilkan, yaitu sebesar lebih kurang 1900 yang berada pada rentang 0 sampai kurang lebih 100 generasi.

Jika dibandingkan dengan penelitian sebelumnya yang menggunakan *backtracking algorithm*, waktu yang dibutuhkan untuk menyelesaikan semua pekerjaan adalah 1662 detik. Dari hasil tersebut, dapat dilihat bahwa metode penjadwalan menggunakan *genetic algorithm* menghasilkan solusi yang lebih optimal, yaitu dapat menghemat 7.64% waktu penyelesaian.

Dengan demikian, dapat dikatakan penerapan *genetic algorithm* cukup optimal jika dibandingkan dengan memakai *backtracking algorithm*.

KESIMPULAN

Berdasarkan dari hasil penelitian dan diskusi yang kami lakukan, dalam permasalahan penjadwalan job shop, *best makespan* yang dihasilkan menggunakan *genetic algorithm* yaitu 1535 detik atau 25.58 menit, dimana lebih cepat 127 detik atau 2.1 menit daripada algoritma *backtracking*. Sehingga, dapat disimpulkan bahwa penggunaan *Genetic Algorithm* dinilai cukup efektif dan efisien dalam menangani permasalahan job shop dibandingkan dengan memakai algoritma *backtracking*.

LINK GITHUB DAN VIDEO

GitHub : sisdas-P1-K5-GA-github
Video : sisdas-P1-K5-GA-video

REFERENSI

[1] Dimyati TT, Taroepratjeka H, Gani AZ, Bahagia SN, Toha IS. 1999. Model optimasi untuk integrasi alokasi produksi dengan penjadwalan operasi *jobshop* dan perencanaan kapasitas. *Jurnal Teknik dan Manajemen Industri* [internet]. 19(1):17-28. Tersedia pada: http://repository.unpas.ac.id/28441/.

[2] Guo ZX, Wong WK, Leung SYS, Fan JT, Chan SF. 2006. Mathematical model and genetic optimization for the job shop scheduling problem in a mixed- and multi-product assembly environment: A case study based on the apparel industry. *Computers & Industrial Engineering* [internet]. 50(3):202-219. Tersedia pada: https://www.sciencedirect.com/science/article/abs/pii/S0360835206000271. DOI: 10.1016/j.cie.2006.03.003.

[3] Husbands P. 1994. Genetic algorithms for scheduling. *AISB Quarterly* [internet]. 89:38-45. Tersedia pada: https://www.staff.ncl.ac.uk/chris.hicks/webpapers/husbands.pdf.

[4] Lin SC, Goodman ED, Punch WF III. 1997. A genetic algorithm approach to dynamic job shop scheduling problems. *Proceedings of the 7th International Conference on Genetic Algorithms* (ICGA); 1997 Juli 19; East Lansing. East Lansing (USA): ICGA. hlm 481-488. Tersedia pada: http://garage.cse.msu.edu/papers/GARAGe97-02-08.pdf.

[5] Nugraha RF. 2018. Sistem penjadwalan otomatis kuliah mahasiswa (Universitas Muhammadiyah Surakarta) [skripsi]. Surakarta (ID): Fakultas Komunikasi dan Informatika, Universitas Muhammadiyah Surakarta. Tersedia pada: http://eprints.ums.ac.id/64403/.

[6] Setiawan JY, Herwindiati DE, Sutrisno T. 2019. Algoritma genetika dengan *roulette wheel selection* dan *arithmetic crossover* untuk pengelompokan. *Jurnal Ilmu Komputer dan Sistem Informasi (JIKSI)* [internet]. 7(1):58-64. Tersedia pada: https://journal.untar.ac.id/index.php/jiksi/article/view/5882.