ALGORITMA FIREFLY (FA) UNTUK MENYELESAIKAN *RESOURCE CONTRAINED PROJECT SCHEDULING PROBLEM* (RCPSP)

Rizqia Wildana Zulfa[1,a)], Vita Kusumasari[2)], Desi Rahmadani[3)]
[1,2,3)] *Jurusan Matematika FMIPA, Universitas Negeri Malang*
[a)]*rwildanazulfa@gmail.com*

Abstrak

Resource Constrained Project Scheduling Problem (RCPSP) merupakan masalah optimasi untuk menjadwalkan kegiatan proyek yang harus memenuhi *precedence constrain* dan *resource constrains* yang bertujuan untuk meminimalkan waktu penyelesaian proyek (*makespan*). Pada penelitian ini dilakukan perhitungan terhadap data 8 aktivitas dengan satu jenis sumber daya dan 32 aktivitas dengan empat jenis sumber daya untuk mendapatkan *makespan* yang optimal dengan kendala RCPSP yang ditetapkan berdasarkan Algoritma *Firefly* (FA) dan dilakukan perbandingan hasil akhir dengan Algoritma *Cukcoo Search* (CS) dan Algoritma *Ant Colony Optimization* (ACO). Perhitungan menggunakan data 8 aktivitas dengan FA, ACO, dan CS diperoleh *makespan* berturut-turut adalah 22, 28, dan 23 satuan waktu. Sedangkan perhitungan menggunakan data 32 aktivitas dengan FA, ACO, dan CS diperoleh *makespan* berturut-turut adalah 38, 58, dan 47 satuan waktu. Berdasarkan hasil perhitungan tersebut menunjukkan bahwa Algoritma *Firefly* (FA) menghasilkan *makespan* yang lebih baik dari Algoritma *Cuckoo Search* dan *Ant Colony Opimization*.

Kata kunci: penjadwalan, *resource constrained project scheduling problem, algoritma firefly*

PENDAHULUAN

Penjadwalan proyek merupakan faktor kunci dalam manajemen proyek yang sukses [1]. Sebuah proyek terdiri dari serangkain aktivitas yang harus dilakukan dengan serangkaian kendala prioritas [2]. Pendekatan penjadwalan proyek secara klasik seperti *Critical Path Method* (CPM) dan *Program Evaluation and Riview Technique* (PERT) yang hanya berfokus pada hubungan ketergantungan aktivitas dan mematuhi *precedence constrain* dengan menggunakan asumsi ketersediaan sumber daya yang tidak terbatas, tetapi dalam kenyataannya manajer proyek sering mengalami kesulitan dalam menjadwalkan suatu proyek dikarenakan keterbatasan sumber daya yang tersedia. Kemudian, jika dipaksakan adanya ketersedian sumber daya tambahan maka akan menyebabkan membesarnya anggaran biaya proyek.

Resource Constrained Project Scheduling Problem (RCPSP) merupakan masalah optimasi untuk menjadwalkan kegiatan proyek yang harus memenuhi *precedence constrain* dan *resource constrains*. *Precedence constrain* merupakan suatu kendala dimana aktivitas tidak dapat dimulai sebelum semua pendahulunya selesai dikerjakan. Sedangkan *resource constrains* merupakan suatu kendala dimana sumber daya yang diperlukan oleh setiap aktivitas pada satuan unit waktu tidak melebihi sumber daya yang tersedia. Fungsi tujuan dari RCPSP yaitu meminimalkan waktu penyelesaian proyek (*makespan*) dibawah kendala ketersediaan sumber daya yang terbatas dan hubungan

prioritas aktivitas *finish-to-start* [3]. Berbagai algoritma telah digunakan dalam menyelesaikan *Resource Constrained Project Scheduling Problem* (RCPSP) diantaranya adalah *Ant Colony Optimization* [4] dan Algoritma *Genetic* (GA) [5]. Pendekatan dengan menggunakan *Ant Colony Optimization* menggunakan data proyek dengan kisaran empat hingga sepuluh kegiatan menghasilkan solusi yang mendekati optimal dengan presentase total 63,64%.

Menurut Davis [6], masalah penjadwalan proyek dengan *job shop* mempunyai kesamaan yang kuat. *Job shop* merupakan masalah penjadwalan operasi mesin-mesin dengan tujuan memperoleh waktu penyelesaian seluruh pekerjaan yang minimum. Penelitian tentang penjadwalan *job shop* sudah banyak dilakukan, salah satunya [7] menggunakan Algoritma *Firefly* (FA). Algoritma *Firefly* dapat menghasilkan solusi optimal dan sangat efisien dengan tingkat keberhasilan yang tinggi dan jauh lebih baik daripada algoritma lainnya seperti ACO dan GA sejauh menyangkut efisiensi dan tingkat keberhasilan [7].

Berdasarkan uraian di atas, akan dibahas penerapan Algoritma *Firefly* (FA) untuk menyelesaikan *Resource Contrained Project Scheduling Problem* (RCPSP). Untuk membuktikan keefektifan Algoritma *Firefly* (FA) pada *Resource Contrained Project Scheduling Problem* (RCPSP) maka Algoritma *Ant Colony Optimization* (ACO) dan Algoritma *Cuckoo Search* (SC) digunakan sebagai pembanding. Hasil perhitungan dari Algoritma *Firefly* (FA) nantinya akan dibandingkan dengan hasil perhitungan dari Algoritma *Ant Colony Optimization* (ACO) dan Algoritma *Cuckoo Search* (SC) pada *Resource Contrained Project Scheduling Problem* (RCPSP). Kemudian beberapa asumsi yang digunakan antara lain: tidak ada penghalang dalam pengerjaan aktivitas dan tidak ada *pre-emption*, artinya apabila suatu aktivitas yan sudah berjalan tidak dapat dihentikan.

METODE

Dalam penelitian ini, RCPSP diselesaikan dengan Algoritma *Firefly* (FA). Algoritma *Firefly* merupakan algoritma yang terinspirasi dari alam yang dikembangkan untuk meniru perilaku kilat kunang-kunang. Algoritma *Firefly* dirumuskan dengan beberapa asumsi dasar yaitu bahwa semua kawanan kunang-kunang mempunyai jenis kelamin tunggal sehingga kunang-kunang yang satu akan tertarik dengan kunang-kunang lainnya tanpa memandang jenis kelaminnya. Daya tarik kunang-kunang sebanding dengan kecerahan dan keduanya berkurang seiring bertambahnya jarak kunang-kunang. Apabila ada dua kunang-kunang yang berkedip, maka kunang-kunang yang kurang terang akan bergerak menuju kunang-kunang yang lebih terang. Jika tidak ada yang lebih terang dari kunang-kunang tertentu, maka kunang-kunang akan bergerak secara acak [8].

Langkah-langkah untuk menerapkan Algoritma *Firefly* [8] adalah sebagai berikut:

1) Inisialisasi parameter.
   Parameter yang digunakan pada Algoritma *Firefly* (FA) yaitu:
   a. m (jumlah populasi)
   b. n (banyak kunang-kunang)
   c. α (parameter pengacakan)
      Parameter pengacakan berupa bilangan random pada interval (0,1]
   d. β_0 (daya tarik pada $r = 0$)
      Daya tarik pada kunang-kunang memiliki nilai yang bervariasi pada interval (0,1]
   e. γ (koefisien penyerapan cahaya)

Koefisien penyerapan cahaya memiliki nilai bervariasi pada interval $(0, \infty)$

f. Maksimum iterasi

2) Membangkitkan populasi awal
   a. Membangkitkan populasi awal *firefly* berupa bilangan real acak sebanyak n aktivitas pada interval $(0,1)$.
   b. Mengurutkan bilangan real acak dari yang terkecil sampai yang terbesar sehigga diperoleh nomor urutan bilangan bulat acak. Menampilkan nomor urutan sesuai dengan posisi bilangan real acak.
   c. Menghitung *makespan* (waktu penyelesaian keseluruhan aktivitas).
3) Menghitung intensitas cahaya pada *firefly* menggunakan persamaan

$$I = \frac{1}{makespan}$$

4) Membandingkan intensitas cahaya *firefly* i dan *firefly* j $(i = 1,2, \dots, m; j = 1,2, \dots, m)$. Jika $I_i > I_j$ maka intensitas *firefly* i dibandingkan dengan *firefly* j yang lain sampai semua *firefly* j selesai dibandingkan dengan *firefly* i, tetapi apabila $I_j > I_i$ maka:
   a. Terjadi pergerakan yaitu *firefly* i bergerak menuju *firefly* j
   b. Setelah pergerakan terjadi kemudian menghitung:
      - Jarak antara dua kunang-kunang i dan j dihitung melalui persamaan
      $$r_{ij} = x_i - x_j$$
      - Daya tarik *firefly* ditentukan oleh intensitas cahaya melalui persamaan
      $$\beta = \beta_0 exp^{(-\gamma r^2)}$$
      - Gerakan *firefly* i terhadap *firefly* yang lebih terang j ditentukan oleh persamaan
      $$x_i^{t+1} = x_i^t + \beta_0 exp^{\left(-\gamma r_{ij}^2\right)}\left(x_j^t - x_i^t\right) + \alpha(rand - 0{,}5)$$
      Dimana β_0 = daya tarik pada jarak $r = 0$
      γ = koefisien penyerapan cahaya
      x_i , x_j = koordinat spasial
      α = parameter pengacakan
      r = jarak antar *firefly*
      t = iterasi,
   c. Setelah mendapatkan intensitas cahaya baru dari *firefly* i kemudian memperbarui nilai intensitas dan melakukan perbandingan intensitas cahaya dengan *firefly* lainnya. Untuk iterasi selanjutnya, jika $I_j' > I_i'$ maka kembali pada proses (a) dan (b) sampai semua *firefly* telah selesai dibandingkan.
5) Menentukan *G-best*. *Firefly* dengan intensitas cahaya tertinggi akan menjadi *G-best*
6) Langakah-langkah Algoritma *Firefly* (FA) diulangi batas iterasi terpenuhi.

HASIL DAN PEMBAHASAN

Pada penelitian ini digunakan dua jenis data, yaitu data yang berisi 8 aktivitas dengan satu jenis sumber daya yang tersedia berjumlah lima unit tiap satuan waktu dan data yang berisi 32 aktivitas, termasuk 2 aktivitas semu berupa *start* dan *end activity* dengan empat jenis sumber daya. Sumber daya jenis R1 berjumlah 12, R2 berjumlah 13, R3 berjumlah 4, dan R4 berjumlah 12 tiap satuan waktu. Data dengan 8 aktivitas [9] dan data dengan 32 aktivitas diambil dari PSPLIB http://www.om-db.wi.tum.de/psplib/. Pada penerapan masing-masing algoritma pada RCPSP dilakukan pengulangan sebanyak 1 kali iterasi.

Selanjutnya akan dipaparkan langkah-langkah perhitungan Algoritma *Firefly* (FA) Pada *Resource Contrained Project Scheduling Problem* (RCPSP) menggunakan data 8 aktivitas.

Pada Tabel 1 diberikan data yang digunakan dalam perhitungan ini.

Tabel 1 Data Set RCPSP 8 Aktivitas

No	Aktivitas	Durasi	Sumber Daya	*Predecessor*	
				1	2
1	1	2	1	-	
2	2	3	3	1	
3	3	4	2	1	
4	4	3	3	2	
5	5	5	2	4	
6	6	7	3	3	
7	7	4	3	6	
8	8	3	2	5	7

Keterangan: *Predecessor* merupakan pendahulu langsung dari sebuah aktivitas. Apabila sebuah aktivitas mempunyai *predecessor*, maka aktivitas tersebut tidak bisa dikerjakan apabila aktivitas pendahulunya belum selesai dikerjakan. Aktivitas 2 mempunyai pendahulu langsung (*predecessor*) aktivitas 1. Untuk menyelesaikan aktivitas 2 membutuhkan durasi 3 satuan waktu dan sumber daya sebanyak 3, dan seterusnya.

Langkah 1. Inisialisasi parameter
Mengidentifikasi parameter yang digunakan dalam menyelesaika contoh kasus Algoritma *Firefly* (FA) adalah sebagai berikut: pengambilan parameter ditentukan berdasarkan (Gupta & Kushwaha, 2018) dengan ukuran populasi m=3, iterasi 1 dan banyak $firefly = 8$ ditentukan oleh banyak aktivitas dalam proyek. $\alpha = 1$ $\beta_0 = 0{,}1$ dan $\gamma = 0{,}01$.

Langkah 2. Membangkitkan populasi awal
Dalam langkah ini, dilakukan pembangkitkan populasi awal *firefly* berupa bilangan real acak sebanyak n aktivitas pada interval (0,1). Hasil pembangkitan populasi awal *firefly* terdapat pada Tabel 2.

Tabel 2 Populasi Awal *Firefly*

Firefly	1	2	3	4	5	6	7	8
x_1	0.037	0.547	0.953	0.976	0.356	0.906	0.711	0.428
x_2	0.998	0.574	0.176	0.659	0.796	0.979	0.034	0.418
x_3	0.033	0.095	0.183	0.445	0.028	0.909	0.791	0.812

Keterangan: posisi populasi x_1 pada aktivitas 1 adalah 0,037 dan seterusnya.

Dalam langkah ini, dilakukan pengurutan bilangan acak dari yang terkecil sampai yang terbesar. Hasil pengurutan elemen populasi *firefly* terdapat pada Tabel 3.

Tabel 3 Pengurutan Populasi *Firefly*

Firefly	1	2	3	4	5	6	7	8
x_1	0.037	0.356	0.428	0.547	0.711	0.906	0.953	0.976
	1	5	8	2	7	6	3	4
x_2	0.034	0.176	0.418	0.574	0.659	0.796	0.979	0.998
	7	3	8	2	4	5	6	1
x_3	0.028	0.033	0.095	0.183	0.445	0.791	0.812	0.909
	5	1	2	3	4	7	8	6

Keterangan: Pada populasi firefly x_2 aktivitas 7 menempati posisi 1 dengan nilai 0,034 dan seterusnya.

Dalam langkah ini, *makespan* diperoleh dari perhitungan durasi penyelesaian keseluruhan aktivitas yang disesuaikan dengan *precedence constrain* dan *resource constrains.* Untuk mengurutkan aktivitas sesuai dengan *precedence constrain* dan *resource constrains* dilakukan dengan mengurutkan solusi aktivitas pada Tabel 3. Pencarian *makespan* disajikan pada Tabel 4.

Tabel 4. Pencarian *Makespan*

Sarang	Urutan aktivitas yang terbentuk								Makespan/$f(x_1)$
x_1	1	2	3	6	7	4	5	8	28
ES	0	2	2	6	13	17	20	25	Satuan waktu
x_2	1	3	2	4	5	6	7	8	22
ES	0	2	2	5	8	8	15	19	Satuan waktu
x_3	1	2	3	4	5	6	7	8	22
ES	0	2	2	5	8	8	15	19	Satuan waktu

Langkah 3. Menghitung intensitas cahaya pada *firefly*
Dalam langkah ini, dilakukan perhitungan intensitas cahaya pada *firefly* dengan menggunakan persamaan $I = \frac{1}{makespan}$. Hasil perhitungan intensitas cahaya terdapat pada Tabel 5.

Tabel 5. Intensitas Cahaya

Firefly	Intensitas Cahaya
x_1	0.035
x_2	0.045
x_3	0.045

Langkah 4. Membandingkan intensitas cahaya
Dalam langkah ini, dilakukan perbandingan intensitas cahaya antara *firefly* i dan j

Ketika $i = 1$

- $I_2 > I_1$, maka terjadi pergerakan *firefly* x_1 menuju *firefly* x_2

Dengan perhitungan menggunakan $x^{t+1} = x^t + \beta\ exp^{(-\gamma r^2)}(x^t - x^t) + \alpha(rand - 0{,}5)$;
$\alpha = 1$ $\beta_0 = 0{,}1$ dan $\gamma = 0{,}01$ diperoleh nilai pergerakan *firefly* dan sudah di urutkan

dari terkecil ke terbesar pada Tabel 6.

Tabel 6. Pergerakan *Firefly* x_1 Menuju *Firefly* x_2

firefly	1	2	3	4	5	6	7	8
x_1^1	0,003	0,561	0,615	0,639	0,750	0,766	0,882	1,281
	5	3	1	7	8	4	2	6

Dari perhitungan diperoleh *makespan* 22 satuan waktu dan intensitas cahaya 0,045.

- $I_3 > I_1$, maka terjadi pergerakan *firefly* x_1 menuju *firefly* x_3

Dengan perhitungan menggunakan $x^{t+1} = x^t + \beta\ exp^{(-\gamma r^2)}(x^t - x^t) + \alpha(rand - 0{,}5)$;
$\alpha = 1\ \beta_0 = 0{,}1$ dan $\gamma = 0{,}01$ diperoleh nilai pergerakan firefly dan sudah di urutkan dari terkecil ke terbesar pada Tabel 7.

Tabel 7. Pergerakan *Firefly* x_1 Menuju *Firefly* x_3

firefly	1	2	3	4	5	6	7	8
x_1^1	0,2795	0,3814	0,4617	0,4793	0,4887	0,6359	0,7202	1,0332
	1	8	6	7	5	2	4	3

Dari perhitungan diperoleh *makespan* 22 satuan waktu dan intensitas cahaya 0,045.
Dari pergerakan *firefly* x_1 menuju x_2 dan x_3 didapatkan *G- best* dengan *makespan* 22 satuan waktu.

Ketika $i = 2$
Karena x_2 mempunyai intensitas cahaya paling besar, sehingga x_2 hanya bergerak secara random.

Dengan perhitungan menggunakan $x^{t+1} = x^t + \beta\ exp^{(-\gamma r^2)}(x^t - x^t) + \alpha(rand - 0{,}5)$; $\alpha = 1\ \beta_0 = 0{,}1$ dan $\gamma = 0{,}01$ diperoleh nilai pergerakan *firefly* dan sudah di urutkan dari terkecil ke terbesar pada Tabel 8.

Tabel 8. Pergerakan *Firefly* x_2 secara random

firefly	1	2	3	4	5	6	7	8
x_1^1	0,185	0,280	0,355	0,366	0,512	1,033	1,246	1,470
	7	4	2	5	3	8	1	6

Dari pergerakan *firefly* x_2 secara random didapatkan *G-best* dengan *makespan* 22 satuan waktu.

Ketika $i = 3$
Karena x_3 mempunyai intensitas cahaya paling besar, sehingga x_3 hanya bergerak secara random.

Dengan perhitungan menggunakan $x^{t+1} = x^t + \beta\ exp^{(-\gamma r^2)}(x^t - x^t) + \alpha(rand - 0{,}5)$; $\alpha = 1\ \beta_0 = 0{,}1$ dan $\gamma = 0{,}01$ diperoleh nilai pergerakan *firefly* dan sudah di urutkan dari terkecil ke terbesar pada Tabel 9.

Tabel 9. Pergerakan *Firefly* x_3 secara random

firefly	1	2	3	4	5	6	7	8
x_1^1	-0,358	-0,286	-0,245	0,293	0,532	0,605	0,783	1,470
	1	5	2	3	6	7	4	8

Dari pergerakan *firefly* x_3 secara random didapatkan *G-best* dengan *makespan* 28 satuan waktu.

Langkah 5. Penentuan *G-best*
Dalam langkah ini, menentukan *G-best* yaitu *firefly* dengan intensitas cahaya terbesar (*firefly* dengan *makespan* terkecil). *Firefly* yang menjadi *G-best* yaitu x_1 dan x_2 dengan *makespan* 22 satuan waktu.

Langkah 6. Kondisi Optimum
Dalam langkah ini, kondisi optimum tercapai ketika semua langkah-langkah Algoritma *Firefly* (FA) telah dilewati dan mencapai iterasi maksimum yaitu 1 iterasi.

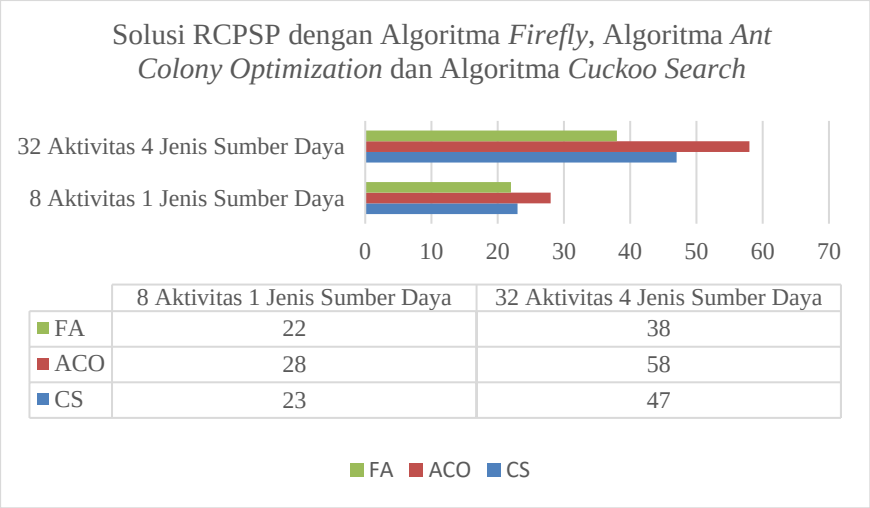
Hasil perhitungan *makespan* untuk ketiga algoritma yang digunakan untuk setiap data set terlihat pada Tabel 10 dan Tabel 11.

Tabel 10. Perbandingan *makespan* dengan Algoritma *Firefly* (FA), Algoritma ACO dan Algoritma *Cuckoo Search* (CS) dengan Data 8 Aktivitas

Algoritma	*Makespan* (satuan waktu)
Algoritma *Firefly* (FA)	22
Algoritma *Ant Colony Optimization* (ACO)	28
Algoritma *Cuckoo Search* (CS)	23

Tabel 11. Perbandingan *makespan* dengan Algoritma *Firefly* (FA), Algoritma ACO, dan Algoritma *Cuckoo Search* (CS) dengan Data 32 Aktivitas

Algoritma	*Makespan* (satuan waktu)
Algoritma *Firefly* (FA)	38
Algoritma *Ant Colony Optimization* (ACO)	58
Algoritma *Cuckoo Search* (CS)	47



	8 Aktivitas 1 Jenis Sumber Daya	32 Aktivitas 4 Jenis Sumber Daya
FA	22	38
ACO	28	58
CS	23	47

Gambar 1. Solusi RCPSP dengan Algoritma *Firefly* (FA), Algoritma *Ant Colony Optimization* (ACO), dan Algoritma *Cuckoo Search* (CS)

Pada Gambar 1 disajikan grafik perbandingan solusi optimal pada RCPSP dengan perhitungan menggunakan Algoritma *Firefly* (FA), Algoritma *Ant Colony Optimization* (ACO) dan Algoritma *Cuckoo Search* (CS). Solusi optimal dari masing-masing algoritma diperoleh dari waktu penyelesaian proyek secara keseluruhan. Pada grafik warna abu-abu menunjukkan hasil optimal dengan Algoritma *Firefly*, warna orange menunjukkan hasil optimal dengan Algoritma *Ant Colony Optimization*, dan warna biru menunjukkan hasil optimal dengan Algoritma *Cuckoo Search*. Untuk data dengan 8 aktivitas dan satu sumber daya penyelesaian dengan FA, ACO, CS menghasilkan solusi optimal (*makespan*) secara berturut-turut adalah 22, 28, dan 23 satuan waktu. Sedangkan, data dengan 32 aktivitas dan 4 sumber daya yang tersedia dengan penyelesaian menggunakan FA,ACO, CS menghasilkan solusi optimal (*makespan*) secara berturut-turut adalah 38, 58, dan 47 satuan waktu. Sehingga, Algoritma *Firefly* menghasilkan *makespan* yang lebih minimum untuk menyelesaiakan RCPSP dibandingkan dengan Algoritma *Cuckoo Search* dan *Ant Colony Optimization*.

KESIMPULAN DAN SARAN

Berdasarkan hasil dan pembahasan yang telah diuraikan diperoleh kesimpulan bahwa Algoritma *Firefly* (FA) dapat diterapkan untuk menyelesaikan permasalahan *Resource Contrained Project Scheduling Problem* (RCPSP) dengan langkah- langkah yaitu menyiapkan data yang sesuai dan inisialisasi parameter, membangkitkan populasi awal *firefly* (FA) dengan membangkitkan blangan real acak pada interval (0,1), mengurutkan bilangan acak, menghitung *makespan*, menghitung intensitas cahaya, membandingkan intensitas cahaya, menentukan *G-best*, kemudian mengulangi proses membandingkan intensitas cahaya sampai menentukan urutan aktivitas terbaik hingga maksimum iterasi terpenuhi. Analisis perbandingan pada penerapan Algoritma *Firefly* (FA), Algoritma *Ant Colony Optimization* (ACO) dan Algoritma *Cuckoo Search* (CS) menghasilkan urutan aktivitas yang berbeda. Algoritma *Firefly* (FA) menghasilkan *makespan* yang lebih minimum yaitu 22 satuan waktu dibandingkan dengan *Ant Colony Optimization* (ACO) yaitu 28 satuan waktu dan Algoritma *Cuckoo Search* (CS) yaitu 23 satuan waktu untuk data dengan 8 aktivitas dengan 1 jenis sumber daya dan sumber daya yang tersedia berjumlah 5. Sedangkan Algoritma *Firefly* (FA) menghasilkan *makespan* Algoritma *Firefly* (FA) menghasilkan *makespan* yang lebih minimum yaitu 38 satuan waktu

dibandingkan dengan *Ant Colony Optimization* (ACO) yaitu 58 satuan waktu dan Algoritma *Cuckoo Search* (CS) yaitu 47 satuan waktu untuk data dengan 32 aktivitas dengan 4 jenis sumber daya dan sumber daya yang tersedia yaitu sumber daya jenis R1 berjumlah 12, R2 berjumlah 13, R3 berjumlah 4, dan R4 berjumlah 12. Kesempatan untuk meneliti kajian serupa masih besar, mengingat studi *Resource Constrained Project Scheduling Problem* (RCPSP) masih tergolong baru dan belum banyak algoritma yang diterapkan dalam masalah ini.

DAFTAR RUJUKAN

[1] S. Zareei, "Project scheduling for constructing biogas plant using critical path method," *Renew. Sustain. Energy Rev.*, vol. 81, pp. 756–759, Jan. 2018, doi: 10.1016/j.rser.2017.08.025.
[2] L. Bianco, M. Caramia, and S. Giordani, "A chance constrained optimization approach for resource unconstrained project scheduling with uncertainty in activity execution intensity," *Comput. Ind. Eng.*, vol. 128, pp. 831–836, Feb. 2019, doi: 10.1016/j.cie.2018.11.053.
[3] M. Tritschler, A. Naber, and R. Kolisch, "A hybrid metaheuristic for resource-constrained project scheduling with flexible resource profiles," *Eur. J. Oper. Res.*, vol. 262, no. 1, pp. 262–273, Oct. 2017, doi: 10.1016/j.ejor.2017.03.006.
[4] N. A. Savitri, "Resource-constrained project scheduling with ant colony optimization algorithm," *J. Civ. Eng.*, vol. 35, no. 2, p. 34, Dec. 2020, doi: 10.12962/j20861206.v35i2.8115.
[5] E. N. Goncharov and V. V. Leonov, "Genetic algorithm for the resource-constrained project scheduling problem," *Autom. Remote Control*, vol. 78, no. 6, pp. 1101–1114, Jun. 2017, doi: 10.1134/S0005117917060108.
[6] E.R. Fitriyani, "Penyelesaian resource-constrained project scheduling problem menggunakan algoritma cat swarm optimization," Universitas Negeri Semarang, 2017.
[7] A. Gupta and S. S. Kushwaha, "An enhanced firefly algorithm approach for solving a flexible job-shop scheduling problem," *Natl. J. Multidiscip. Res. Dev.*, vol. 3, no. 1, pp. 463–468, Jan. 2018.
[8] A. M. Altabeeb, A. M. Mohsen, and A. Ghallab, "An improved hybrid firefly algorithm for capacitated vehicle routing problem," *Appl. Soft Comput.*, vol. 84, p. 105728, Nov. 2019, doi: 10.1016/j.asoc.2019.105728.
[9] R. I. Putra, "Penerapan Algoritma Harmony Search Pada Resource-Constrained Project Scheduling Problem (RCPSP)," Universitas Negeri Malang, 2015.