

## OPTIMASI *RESOURCE CONSTRAINED PROJECT SCHEDULING PROBLEM* MENGGUNAKAN METODE *GENETIC ALGORITHM* DAN *SYMBIOTIC ORGANISMS SEARCH*

Evan Filbert Sikomena<sup>1</sup>, I Gede Agus Widyadana<sup>2</sup>, Doddy Prayogo<sup>3</sup>

<sup>1</sup> Mahasiswa Program Studi Magister Teknik Sipil, Universitas Kristen Petra, Surabaya, Indonesia

<sup>2,3</sup> Dosen Program Studi Magister Teknik Sipil, Universitas Kristen Petra, Surabaya, Indonesia

<sup>1</sup> b21180008@john.petra.ac.id, <sup>2</sup> gede@petra.ac.id, <sup>3</sup> prayogo@petra.ac.id

**ABSTRAK:** *Resource constrained project scheduling problem* (RCPSP) merupakan gabungan dari permasalahan sumber daya yang terbatas dan juga permasalahan perencanaan penjadwalan proyek. Penting untuk mengatur atau mengalokasikan sumber daya yang terbatas guna menghasilkan waktu yang optimal. Untuk itu dilakukan penelitian lebih mendalam untuk mengetahui metode mana yang baik dalam menyelesaikan RCPSP. Ada lima jenis proyek dengan karakteristik berbeda-beda serta jumlah *item* pekerjaan beragam yang digunakan sebagai *study case*. Penelitian ini membandingkan *symbiotic organisms search* (SOS) dan *genetic algorithm* (GA) untuk melihat mana yang memberikan waktu yang optimal dengan sumber daya yang terbatas. Kelima proyek menggunakan parameter yang sama yaitu populasi 50 dan iterasi 500 kali untuk SOS dan GA. Hasil dari penelitian ini menyimpulkan bahwa SOS memberikan waktu yang lebih optimal dibandingkan dengan GA. Dari hasil optimasi, SOS menghasilkan rata-rata penurunan waktu sebanyak 30.35% sedangkan GA menghasilkan rata-rata penurunan waktu sebanyak 25.39%.

Kata kunci: *resource constrained project scheduling problem*, optimasi, metaheuristik, *symbiotic organisms search*, *genetic algorithm*

**ABSTRACT:** *Resource constrained project scheduling problem* (RCPSP) is a combination of limited resource problems and project scheduling planning problems. It is important to manage or allocate limited resources in order to produce optimal time. This paper presents a research to find out which methods work well for completing the RCPSP. There are five types of projects with different characteristics and a variety of work items those are used as study cases. This study compares the *symbiotic organisms search* (SOS) and *genetic algorithm* (GA) to see which one gives the optimal time with limited resources. The five projects used the same parameters, such as population 50 and iterations 500 times for SOS and GA. The results of this study concluded that SOS provides a more optimal time than GA. From the optimization results, SOS produces an average time reduction of 30.35% while GA produces an average time reduction of 25.39%.

Keywords: *resource constrained project scheduling problem*, optimization, metaheuristic, *symbiotic organisms search*, *genetic algorithm*

## 1. PENDAHULUAN

Mengatur atau mengalokasikan sumber daya yang terbatas merupakan hal yang penting untuk direncanakan dengan tujuan untuk menghasilkan produk dengan standar yang telah ditentukan (Soeharto, 1999). Dalam perkembangan jaman yang semakin maju ini, proyek-proyek besar dan kompleks pun semakin banyak. Di dalam pengerjaan suatu proyek pasti membutuhkan optimasi untuk mengatur sumber daya (bahan dan material, tenaga kerja, serta peralatan-peralatan yang digunakan). Proyek sendiri memiliki jangka waktu atau batasan waktu untuk diselesaikan sehingga dibutuhkan metode untuk mengatur sumber daya agar dapat meminimalisir *waste* dan meningkatkan *profit* serta dapat menyelesaikan proyek tersebut sesuai dengan waktu yang ditentukan. Permasalahan sumber daya yang terbatas dan juga permasalahan perencanaan penjadwalan proyek ini biasanya disebut dengan *resource constrained project scheduling problem* (RCPSP) (Santosa & Willy, 2011). RCPSP merupakan salah satu masalah yang dikembangkan untuk mendekati keadaan realita lapangan (Noranita, 2003). Proyek sendiri memiliki sejumlah aktivitas-aktivitas didalamnya dimana setiap aktivitas memiliki durasi dan jumlah sumber daya yang berbeda-beda. RCPSP sendiri merupakan masalah optimasi yang sulit untuk diselesaikan. Seiring dengan perjalannya waktu, banyak penelitian yang menggunakan metode heuristik dan metode metaheuristik untuk menyelesaikan kasus RCPSP.

Pada penelitian sebelumnya yang dilakukan oleh Chu dan Tsai (2006), telah ditunjukkan bahwa menggunakan algoritma *cat swarm optimization* (CSO) lebih efisien dari pada algoritma *particle swarm optimization* untuk menyelesaikan kasus RCPSP. CSO juga dapat memberikan hasil yang lebih baik dari segi jumlah iterasinya (Dhanasaputra & Santoso, 2010). Kemudian penelitian lanjutan tentang CSO dilakukan oleh Fitriyani (2017) dan mendapat kesimpulan bahwa CSO menghasilkan solusi yang optimal untuk menyelesaikan kasus RCPSP. Kekurangan dari CSO sendiri adalah harus dilakukan simulasi yang berulang untuk memastikan solusi yang didapatkan merupakan solusi yang optimal.

Pada penelitian ini akan dilakukan hal yang sama yaitu menyelesaikan kasus RCPSP tetapi menggunakan *genetic algorithm* yang dapat memaksimalkan mekanisme *updating* kromosom sehingga membuat *genetic algorithm* mampu keluar dari solusi yang bersifat lokal optimal (Fwa, et al., 1996) dan metode yang ditemukan oleh Cheng dan Prayogo (2014) yaitu metode *symbiotic organisms search* (SOS). SOS sendiri sudah terbukti baik dalam menyelesaikan permasalahan optimasi tetapi belum pernah digunakan untuk menyelesaikan kasus RCPSP sehingga metode SOS ini yang menjadi hal baru yang dicoba untuk menyelesaikan kasus RCPSP ini. Pada penelitian ini yang dioptimalkan adalah waktu dengan kendala sumber daya yang ditentukan.

## 2. LANDASAN TEORI

### 2.1 *Resource Constrained Project Scheduling Problem* (RCPSP)

*Resource constrained project scheduling problem* (RCPSP) merupakan permasalahan penjadwalan proyek dimana harus memenuhi *precedence constraints* dan *resource constraints* (Putra, 2013). *Precedence constraints* merupakan keadaan dimana aktivitas terdahulu sudah dijadwalkan sebelum aktivitas setelahnya dijadwalkan. Sedangkan *resource constraints* merupakan permasalahan untuk mengatur sumber daya dimana sumber daya

yang dibutuhkan tidak boleh melebihi sumber daya yang tersedia. Menurut Setiawan dan Santoso (2010) sumber daya yang dibutuhkan oleh setiap aktivitas proyek dapat bertipe satu jenis atau lebih dari satu jenis. Hal ini tergantung bisa atau tidaknya suatu sumber daya diperbarui.

Berikut ini merupakan rumus matematika dari RCPSP dengan tujuan meminimalkan durasi proyek (Talbot, 1982):

$$\min\{\max f_i \mid i = 1, 2, \dots, N\} \dots \dots \dots (2.1)$$

$$\text{Subject to} \quad : f_i - f_j \geq d_i \dots \dots \dots (2.2)$$

$$\begin{aligned} & \square_j \in P_i; i = 1, 2, \dots, N \\ & \sum_{A_t} r_{ik} \leq R_k \end{aligned} \dots \dots \dots (2.3)$$

$$k = 1, 2, \dots, K; t = s_1, s_2, \dots, s_N$$

- di mana
- $N$  = jumlah aktivitas yang terdapat pada proyek
  - $f_i$  = waktu penyelesaian aktivitas  $i$  ( $i = 1, 2, \dots, N$ )
  - $d_i$  = durasi aktivitas  $i$
  - $P_i$  = *predecessor* aktivitas  $i$
  - $R_k$  = jumlah sumber daya tipe  $k$  yang tersedia ( $k = 1, 2, \dots, K$ )
  - $K$  = jumlah tipe sumber daya
  - $r_{ik}$  = jumlah sumber daya tipe  $k$  yang diperlukan oleh aktivitas  $i$
  - $A_t$  = sekelompok aktivitas yang berjalan pada waktu ke- $t$
  - $s_i$  =  $f_i - d_i$ ;  $s_i$  = waktu mulai aktivitas  $i$

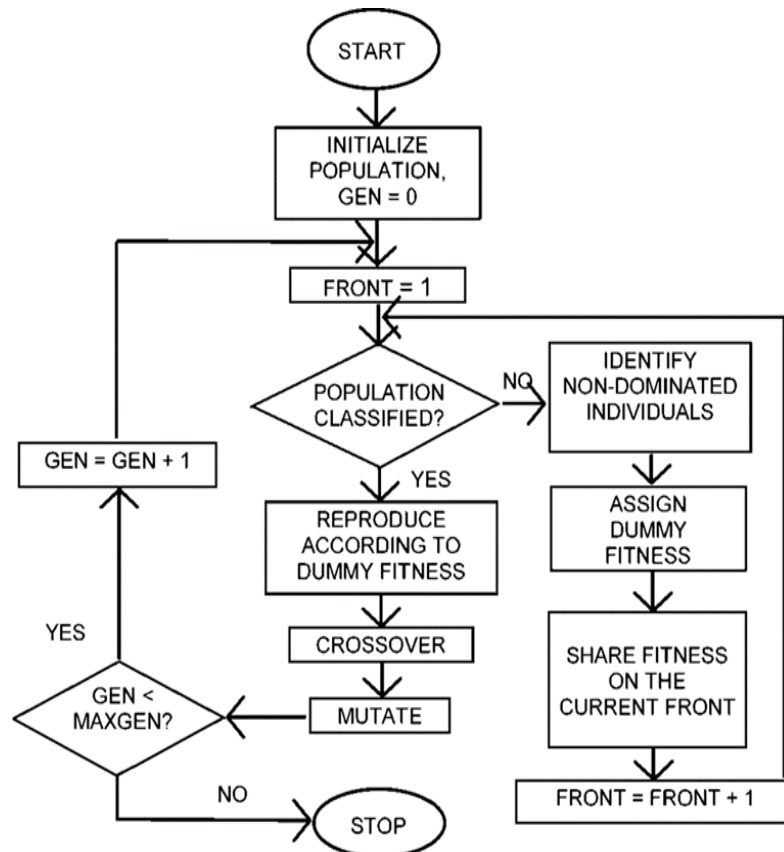
Persamaan (2.1) merepresentasikan fungsi tujuan dari RCPSP. Persamaan (2.2) merepresentasikan *precedence constraints*. Persamaan (2.3) merepresentasikan *resource constraints*.

## 2.2 Genetic Algorithm

*Genetic algorithm* merupakan teknik optimasi yang diperkenalkan oleh John Holland dalam buku yang berjudul “*Adaption in natural and artificial systems*” yang diterbitkan pada tahun 1975. *Genetic algorithm* berbeda dengan teknik optimasi pada umumnya, dimana pada kondisi *genetic algorithm* diawali dengan *setting* awal solusi acak yang disebut populasi. Tiap individu dalam populasi di sebut dengan kromosom, yang merepresentasikan suatu solusi atas permasalahan. Kromosom akan berevolusi melalui iterasi berkelanjutan yang disebut generasi. Dikutip dari Yulianto (2013), Individu yang memiliki kinerja baik dapat mempertahankan eksistensinya dan mendapatkan pasangan untuk melahirkan solusi baru yang lebih baik. Siklus umum dari *genetic algorithm* ditunjukkan pada Gambar 1. Secara garis besar langkah umum pada *genetic algorithm* adalah sebagai berikut:

- a. Melakukan inialisasi populasi kromosom dengan solusi secara acak (*random*).
- b. Melakukan evaluasi setiap kromosom dalam populasi menggunakan persamaan fungsi evaluasi (*fitness function*).
- c. Memilih sebagai anggota populasi sebagai solusi yang sesuai dengan induknya untuk generasi selanjutnya.

- d. Menciptakan solusi (keturunan) baru dengan mengawinkan solusi dari induknya dengan cara *crossover* dan mutasi.
- e. Membuang atau menghapus anggota populasi lama yang tidak produktif untuk membuat ruang solusi yang baru agar dapat masuk kedalam populasi.
- f. Jika aturan pemberhentian terpenuhi, berhenti dan keluarkan kromosom yang baik. Jika tidak kembali ke langkah c



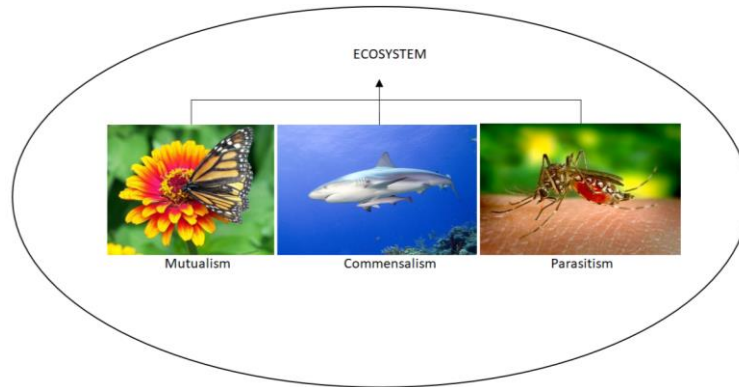
Gambar 1. Siklus Umum *Genetic Algorithm*

Sumber: Bagchi, T. P. (2003). Multiobjective robust design by genetic algorithm. *Materials and Manufacturing Processes*, 18(3), 341-354.

### 2.3 *Symbiotic Organisms Search*

*Symbiotic organisms search* adalah metode optimasi metaheuristik yang melakukan simulasi terhadap macam-macam organisme didalam ekosistem melalui interaksi simbiosis dengan tujuan untuk meningkatkan tingkat kelangsungan hidup dari organisme tersebut (Cheng & Prayogo, 2014). Simbiosis sendiri terbagi menjadi tiga jenis yaitu simbiosis mutualisme, komensalisme dan parasitisme. Setiap variabel solusi diumpamakan sebagai organisme. Organisme akan saling berinteraksi dengan sesama secara acak dan melakukan tiga jenis simbiosis tersebut. Pada Gambar 2 akan diilustrasikan ketiga jenis simbiosis tersebut. Metode SOS ini memiliki tiga fase, yaitu:

1. Fase inialisasi;
2. Fase simbiosis (*mutualism phase*, *commensalism phase*, dan *parasitism phase*);
3. Fase pengulangan hingga mendapatkan hasil terbaik.



Gambar 2. Ilustrasi Simbiosis yang Terjadi antar Makhluk Hidup

### 3. METODE PENELITIAN

#### 3.1 Optimasi Penjadwalan Proyek Menggunakan Metode Metaheuristik

Penelitian ini dilakukan dengan melewati beberapa tahapan, yaitu:

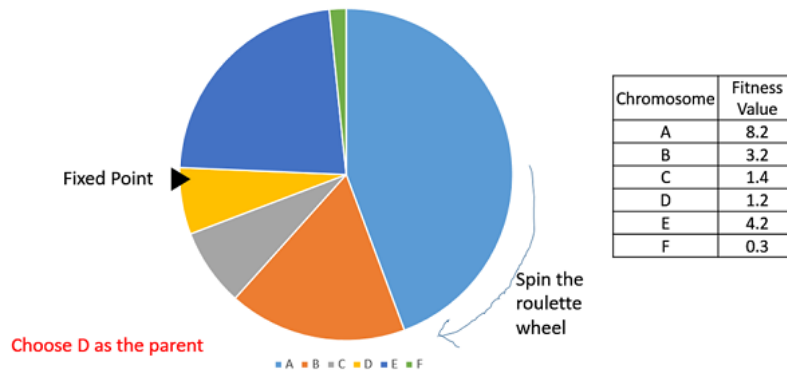
1. Berawal dengan pengumpulan data informasi proyek berupa RAB, *time schedule* dari kontraktor dan wawancara untuk jumlah pekerja.
2. Kemudian dilanjutkan dengan permodelan optimasi metode metaheuristik. Permodelan dimulai dengan menentukan jumlah populasi, merepresentasikan *precedence constraints*, merepresentasikan *resource constraints* dan fungsi objektif.
3. Dilanjutkan dengan pembuatan populasi awal dengan jumlah sumber daya yang diambil sesuai dengan perhitungan teoritis dan juga berdasarkan wawancara dengan kontraktor.
4. Setelah selesai pembuatan populasi awal, dilanjutkan dengan menganalisa jumlah sumber daya dengan cara menghitung fungsi objektif populasi awal, dan kemudian dilanjutkan dengan memodifikasi populasi dengan algoritma metaheuristik. Algoritma metaheuristik yang digunakan adalah *genetic algorithm* dan *symbiotic organisms search*.
5. Setelah populasi baru dibuat, kemudian dilakukan perhitungan untuk fungsi objektif dari populasi baru tersebut. Jika hasil populasi baru tersebut memiliki *precedence constraints* dan *resource constraints* yang belum tepat, maka dilakukan modifikasi terhadap populasi dengan menggunakan algoritma metaheuristik sampai dengan menghitung fungsi objektif dari populasi baru tersebut lagi. Tapi jika hasil populasi awal tersebut sudah memiliki *precedence constraints* dan *resource constraints* maka hasilnya dapat disimpan untuk dilanjutkan.
6. Pada tahap menghitung Fungsi Objektif ( $Z$ ), dilakukan penambahan fungsi bagi hasil yang melanggar *constraints* yang diberikan sehingga pilihan tersebut akan ditolak secara otomatis. Rumus perhitungan fungsi objektif dibentuk menjadi sebagai berikut:

$$Z = \begin{cases} \min(\max f_i + 1000), & r_{ik} > R_k \\ \min(\max f_i) & , \quad r_{ik} \leq R_k \end{cases}$$

7. Setelah semua sudah terpenuhi maka akan didapat hasil yang optimal. Hasil yang optimal dimaksud adalah ketika jumlah sumber daya yang digunakan dapat dimaksimalkan penggunaannya secara efektif dan memiliki waktu penyelesaian proyek yang dapat dipercepat. Jumlah sumber daya yang digunakan pun tidak akan melebihi batas kewajaran dalam suatu item pekerjaan.

### 3.2 Genetic Algorithm

*Genetic algorithm* dilakukan dengan melalui tiga tahapan yaitu *parent selection*, *crossover*, dan mutasi. Untuk *parent selection* yang digunakan GA adalah *roulette wheel selection*. Metode *selection* berbentuk seperti lingkaran yang terbagi-bagi seperti *pizza*. Untuk besar potongan *pizza* ditentukan dari seberapa bagus nilai *fitness* dan ada satu titik yang menjadi titik pemilihan ketika lingkaran tersebut diputar. Untuk ilustrasinya dapat dilihat pada gambar 3.



Gambar 3. Ilustrasi *Roulette Wheel Selection*

Sumber: *Genetic Algorithms – Parent Selection*. (n.d.).

[https://www.tutorialspoint.com/genetic\\_algorithms/genetic\\_algorithms\\_parent\\_selection.htm](https://www.tutorialspoint.com/genetic_algorithms/genetic_algorithms_parent_selection.htm)

Untuk *crossover* yang digunakan adalah metode dua titik dengan probabilitas 0,9 dan mutasi yang digunakan adalah mutasi *uniform* dengan probabilitas 0,01. Untuk populasi yang digunakan adalah 50 dan iterasi yang digunakan adalah 500 kali.

### 3.3 Symbiotic Organisms Search

*Symbiotic organisms search* bermula dengan setiap organisme saling berinteraksi. Interaksi yang dilakukan melewati 3 tahapan yaitu simbiosis mutualisme dilanjutkan dengan simbiosis komensalisme dan diakhiri dengan simbiosis parasitisme. Setiap variabel solusi diumpamakan sebagai organisme. Organisme akan saling berinteraksi dengan sesama secara acak dan melakukan 3 jenis simbiosis tersebut. Untuk parameter yang digunakan SOS adalah populasi 50 dan iterasi 500 kali.

### 3.4 Metode Membandingkan GA dan SOS

Cara membandingkan:

1. Melakukan pengecekan konsistensi  
Dilakukan 40x running untuk kemudian dihitung berapa standar deviasi dan hari optimum dari setiap metode kemudian dibandingkan.
2. Melakukan pengujian dengan *t-test independent*  
Dengan menggunakan *t-test independent* hasil rata-rata hari optimum dibandingkan.
3. Grafik konvergensi  
Dibandingkan kecepatan mencapai lokal optimum dan waktu terjadinya konvergen

## 4. HASIL PENELITIAN

### 4.1 Informasi Proyek dan Perhitungan Sumber Daya Teoritis

Pada penelitian ini digunakan proyek pembangunan gereja yang memiliki karakteristik gedung satu lantai tetapi memiliki ketinggian gedung yg mencapai 10 meter. Informasi *item* pekerjaan proyek, *predecessor*, dan *successor* dari *time schedule* sesungguhnya proyek tersebut. Jumlah sumber daya dihitung dengan menggunakan peraturan menteri pekerjaan umum (Permen PU) Nomor 11/PRT/M/2013 dan beberapa *item* pekerjaan yang dilakukan wawancara langsung dengan kontraktor guna untuk mendapatkan koefisien dari analisa harga satuan yang digunakan kontraktor. Untuk contoh pada *item* pekerjaan pasangan pondasi batu kali digunakan koefisien sebagai berikut,

1 m<sup>2</sup> Pasangan Pondasi Batu Kali:

Pekerja	1.5	OH
Tukang batu	0.75	OH

Volume pasangan batu kali yang ada didalam RAB adalah 157.08 m<sup>2</sup>, sehingga untuk mendapatkan durasi pekerjaan pasangan batu kali tersebut maka 1 m<sup>3</sup> dibagikan dengan koefisien dari Permen PU dan didapatkan produktifitas dari *item* pekerjaan tersebut. Kemudian volume dibagikan dengan produktifitas didapatkan durasi pekerjaan ketika dikerjakan 1 orang, sehingga,

Produktifitas

Pekerja	1	:	1.5	= 0.67
Tukang batu	1	:	0.75	= 1.33

Durasi *item* pekerjaan ketika dikerjakan hanya 1 orang

Pekerja	157.08	:	0.67	= 235.62 hari
Tukang batu	157.08	:	1.33	= 117.81 hari
				353.43 hari ~ 354 hari

### 4.2 Permodelan Data Proyek dan Parameter Setting SOS dan GA

Data *item* pekerjaan yang digunakan dalam permodelan awal adalah *ID item* pekerjaan, *delay*, *predecessor*, durasi ketika dikerjakan satu orang, sumber daya nyata (*real*), sumber daya batas bawah dan sumber daya batas atas. *Delay* adalah kondisi dimana suatu *item* pekerjaan dapat mulai dikerjakan sebelum *predecessor*-nya 100% selesai atau suatu *item* pekerjaan yang dapat dikerjakan beberapa hari setelah *predecessor*-nya 100% selesai. *Predecessor* adalah kondisi dimana suatu *item* pekerjaan dapat dikerjakan setelah pekerjaan pendahulunya selesai. Durasi ketika dikerjakan satu orang merupakan kondisi penting yang menentukan suatu *item* pekerjaan dapat diselesaikan dalam berapa hari dengan jumlah pekerja dan tukang yang secara *random* dilakukan oleh SOS dan GA. Sumber daya nyata (*real*) merupakan jumlah sumber daya pada setiap *item* pekerjaan yang digunakan untuk membuat CPM awal. Sedangkan untuk sumber daya batas bawah dan batas atas digunakan sebagai input didalam optimasi waktu penyelesaian proyek-proyek tersebut. Parameter *setting* yang digunakan SOS dan GA ditunjukkan pada Tabel 1.

Tabel 1. Parameter SOS dan GA

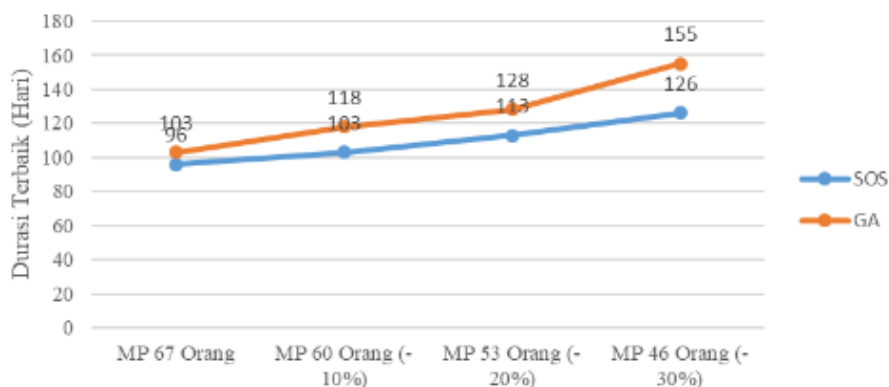
SOS	GA
Populasi = 50	Selection = Roulette
Iterasi = 500	Crossover Fraction = 0.9
	Mutation = Uniform
	Mutation Rate = 0.01
	Populasi = 50
	Iterasi = 500

### 4.3 Perbandingan Hasil Optimasi

Pada Tabel 2 dan Gambar 4 dapat dilihat bahwa SOS menghasilkan nilai durasi terbaik yang lebih optimum untuk kelima gedung dibandingkan GA dengan parameter yang digunakan. Tetapi ketika dibandingkan dengan waktu pencarian selama 500 kali iterasi, GA memiliki pencarian lebih cepat dibandingkan dengan SOS. Dari hasil optimasi tersebut didapatkan rata-rata penurunan waktu yang dihasilkan SOS adalah 30.35% dan rata-rata penurunan waktu yang dihasilkan GA adalah 25.39%.

Tabel 2. Hasil Optimasi SOS dan GA

Nama Proyek	Data	SOS (Hari)				GA (Hari)			
		MP 67 Orang	MP 60 Orang (-10%)	MP 53 Orang (-20%)	MP 46 Orang (-30%)	MP 67 Orang	MP 60 Orang (-10%)	MP 53 Orang (-20%)	MP 46 Orang (-30%)
Gereja (150 hari)	<i>Best</i>	96	103	113	126	103	118	128	155
	<i>Worst</i>	101	108	122	137	117	144	169	275
	<i>Mean</i>	97.90	105.60	118.00	132.30	112.60	130.30	149.10	216.80
	Std Deviasi	1.4491 4	1.5055 5	2.9059 3	3.6224 6	4.6714 3	8.8198 0	12.5383 9	50.1083 3
	Penurunan	36.00%				31.33%			
	Waktu Running	73 menit				62 menit			



Gambar 4. Durasi Terbaik Proyek Gereja



#### 4.4 Uji T-Test Independen

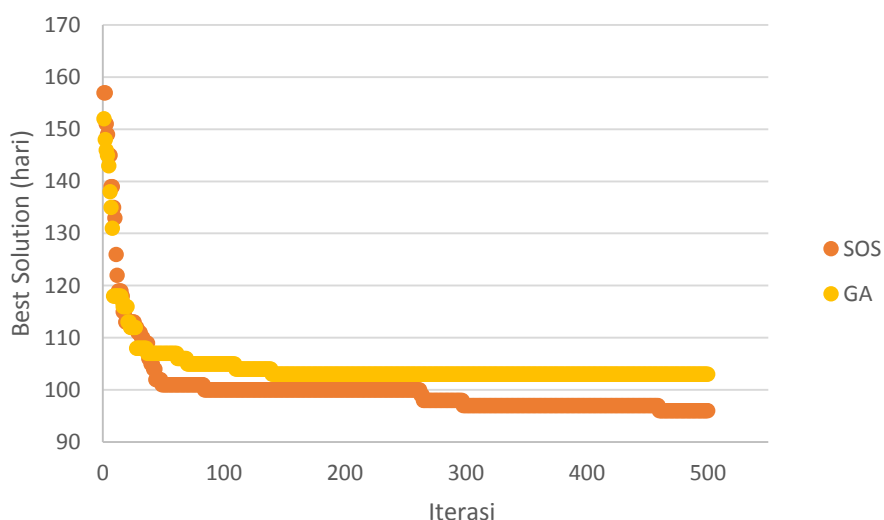
Pada Tabel 3 ditunjukkan bahwa hasil *t-test independent* pada gedung gereja memiliki t-hitung yang lebih besar dari t-tabel sehingga disimpulkan adanya perbedaan dari hasil rata-rata kedua algoritma dan SOS memiliki hasil yang lebih baik dibandingkan GA.

Tabel 3. Perbandingan Uji T-Test Independent

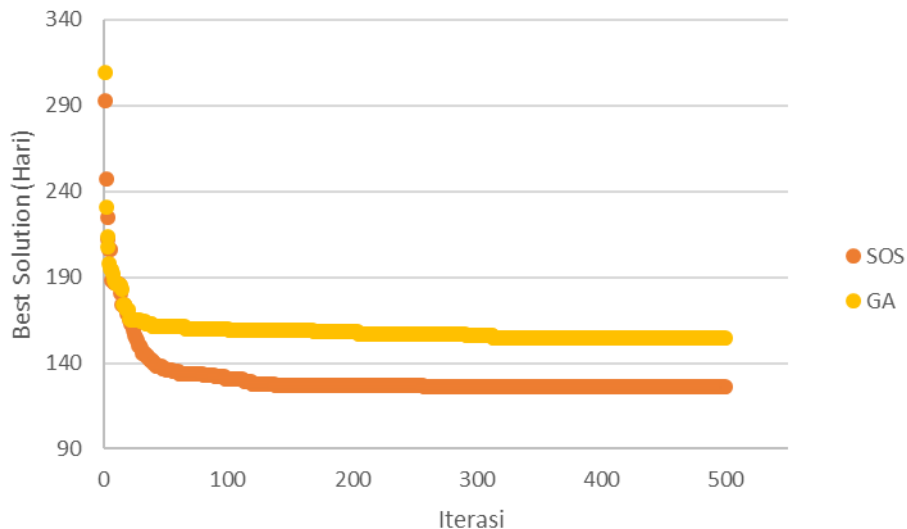
Proyek	Kategori T	Nilai T (5%)	Kesimpulan
Gereja	T-Hitung	9.50	T-Hitung > T-Tabel
	T-Tabel	1.83	

#### 4.5 Grafik Konvergensi

Pada Gambar 5 yang menggunakan maksimum pekerja per hari nyata dan gambar 6 yang menggunakan maksimum pekerja per hari dikurangi 30%, dapat dilihat GA dan SOS sudah mengalami konvergensi. Dari 200 iterasi terakhir tidak ada lagi penurunan yang lebih dari 3% kondisi nyata. Sehingga dapat disimpulkan bahwa sudah tidak ada lagi penurunan yang signifikan dari kedua algoritma tersebut. Dengan penurunan 30% maksimum pekerja per hari dapat dilihat bahwa selisi antara GA dan SOS semakin membesar.



Gambar 5. Diagram Konvergensi SOS dan GA Proyek Gereja (MP Nyata)



Gambar 6. Diagram Konvergensi SOS dan GA Proyek Gereja (-30%)

#### 4.6 Pembahasan Hasil

Dengan parameter yang digunakan dapat disimpulkan bahwa SOS memiliki hasil yang lebih optimal dibandingkan dengan GA, tetapi dari kecepatan untuk menyelesaikan 500 iterasi GA memiliki kecepatan yang lebih baik dari SOS, sehingga jika dicoba menyamakan waktu penyelesaian, GA dapat menyelesaikan lebih banyak iterasi dibandingkan SOS. Dari standar deviasi yang dihasilkan GA dan SOS dapat disimpulkan GA lebih cepat terjebak dilokal optimal dari pada SOS. Dari hasil pengurangan maksimum pekerja 10%, 20% dan 30%, dapat ditemukan bahwa hasil optimum yang diberikan oleh SOS dan GA pun menjadi terbatas.

Perbedaan hasil optimum antara SOS dan GA tidak terlalu besar pada sumber daya nyata dan semakin besar perbedaannya ketika sumber daya yang semakin diperketat, sehingga SOS semakin unggul dibandingkan GA jika alokasi jumlah sumber daya semakin sedikit. Berdasarkan hasil perbandingan *t-test independent* didapatkan bahwa rata-rata waktu penyelesaian optimal antara SOS dan GA berbeda secara signifikan, sehingga SOS memiliki hasil yang lebih baik dari GA pada kasus proyek gereja.

Dengan menggunakan 500 iterasi, SOS dan GA sudah terjebak dalam lokal optimal dan sudah konvergen. Tidak ada lagi penurunan yang signifikan dari 200 iterasi terakhir pada kedua algoritma. Sehingga dapat disimpulkan bahwa penggunaan 500 iterasi sudah cukup untuk mendapatkan hasil optimum pada kedua algoritma.

### 5. KESIMPULAN DAN SARAN

Dari hasil penelitian dan analisis yang dilakukan, dapat disimpulkan bahwa *genetic algorithm* (GA) dan *symbiotic organisms search* (SOS) terbukti mampu memberikan solusi yang lebih baik dan menghasilkan waktu yang optimal dengan sumber daya yang ditentukan dalam kasus *resource constrained project scheduling problem* (RCPSP). Dengan parameter yang digunakan dapat disimpulkan *symbiotic organisms search* (SOS) memberikan hasil optimasi yang lebih baik dengan rata-rata penurunan waktu sebanyak 30.35% sedangkan *genetic algorithm* (GA) memberikan rata-rata penurunan waktu sebanyak 25.39%.

Penelitian ini untuk mengetahui waktu penyelesaian proyek dengan batasan sumber daya. Selanjutnya dapat dibuat penelitian untuk menentukan berapa jumlah sumber daya dan biaya yang optimal dengan target hari penyelesaian yang ditentukan.

## 6. DAFTAR REFERENSI

- Bagchi, T. P. (2003). Multiobjective Robust Design by Genetic Algorithms. *Materials and Manufacturing Processes*, 18(3), 341-354.
- Cheng, M. Y., & Prayogo, D. (2014). Symbiotic Organisms Search: A New Metaheuristic Algorithm. *Computers & Structures*, 139, 98-112.
- Chu, S. C., & P. W. Tsai., (2006). Cat Swarm Optimization. *Proceedings of the 9<sup>th</sup> Pasific Rim International Conference on Artificial Intelligence LNAI 4099*, (pp. 854-858). Berlin: Springer.
- Dhanasaputra, N., & Santosa, B. (2010). Pengembangan Algoritma Cat Swarm Optimization (CSO) untuk Klasifikasi. *Jurnal ITS*, 3.
- Fitriyani, E. R., (2017). *Penyelesaian Resource-Constrained Project Scheduling Problem Menggunakan Algoritma Cat Swarm Optimization*, (Dissertation, Universitas Negeri Semarang).
- Fwa, T. F., Chan, W. T., & Tan, C. Y. (1996). Genetic-Algorithm Programming of Road Maintenance and Rehabilitation. *Journal of Transportation Engineering*, 122(3), 246-253.
- Genetic Algorithm – Parent Selection. (n.d.).  
[https://www.tutorialspoint.com/genetic\\_algorithms/genetic\\_algorithms\\_parent\\_selection.htm](https://www.tutorialspoint.com/genetic_algorithms/genetic_algorithms_parent_selection.htm)
- Noranita, B., (2003). *Penjadwalan Proyek dengan Parameter Waktu Pelaksanaan yang Tidak Pasti dan Batasan Persediaan Sumber Daya*. Unpublished master's thesis, Universitas Gadjah Mada, Yogyakarta.
- Putra, R. I., (2013). Penerapan Algoritma Harmony Search pada Resource Constrained Project Scheduling Problem (rcpsp). *Jurnal Online Jurusan Matematika-Fakultas MIPA UM*.
- Santosa, B. dan Willy. P., (2011). *Metoda Metaheuristik: Konsep dan Implementasi*. Guna Widya, Surabaya.
- Setiawan, A., & Santosa, I. B. (2010). *Penerapan Algoritma Harmony Search dalam Penyelesaian Resource-Constrained Project Scheduling Problem*. Unpublished master's thesis, Institut Teknologi Sepuluh Nopember, Surabaya.
- Soeharto, I. (1999). *Manajemen Proyek (Dari Konseptual sampai Operasional)* (2<sup>nd</sup> ed.). Erlangga, Jakarta.
- Talbot, F. B. (1982). Resource-Constrained Project Scheduling with Time-Resource Tradeoffs: the Nonpreemptive Case. *Management Science*, 28(10), 1197-1210.
- Yulianto, A. (2013). Optimasi Penjadwalan Proyek Menggunakan Cpm dengan Algoritma Genetika pada Studi Kasus Proyek Pembangunan Laboratorium Ekonomi Ubhara Surabaya. *Extrapolasi: Jurnal Teknik Sipil*, 6(02), 35-42.