# BAB III
# METODE PENELITIAN

Bab ini membahas metode yang akan digunakan untuk menyelesaikan masalah penjadwalan proyek. Pembahasan ini mencakup deskripsi masalah, tahap pengumpulan data, model optimasi, dan contoh kasus serta penyelesaiannya.

## 3.1 Deskripsi Masalah

Masalah penjadwalan proyek dengan sumber daya terbatas (RCPSP) merupakan salah satu tantangan utama dalam manajemen proyek yang kompleks. Tujuan utama dari penelitian ini adalah menentukan waktu mulai proyek yang optimal untuk setiap aktivitas proyek sehingga keseluruhan proyek dapat diselesaikan dalam waktu minimum (*makespan*) dengan tetap memenuhi batasan terkait ketergantungan antar-aktivitas dan ketersediaan sumber daya. Dalam penelitian ini, fokus penelitian diarahkan pada proyek konstruksi yang memiliki karakteristik unik dalam penjadwalan aktivitas seperti beberapa pekerjaan harus dilakukan secara berurutan, seperti memasang fondasi sebelum mendirikan dinding, sementara pekerjaan lain dapat dikerjakan secara bersamaan, seperti pemasangan listrik dan pengecatan di ruangan berbeda.

Proyek terdiri dari $n$ aktivitas, termasuk dua aktivitas *dummy*: aktivitas awal ($i = 0$) yang menunjukkan "proyek dimulai" dan aktivitas akhir ($i = n + 1$) yang menandakan "proyek selesai". Setiap aktivitas memiliki durasi tertentu ($d_i$) yang telah diketahui sebelumnya dan membutuhkan sumber daya untuk pelaksanaannya. Sumber daya ini terbagi dalam beberapa tipe ($K$), di mana setiap tipe memiliki kapasitas maksimum ($R_k$) yang tetap selama periode pelaksanaan proyek. Aktivitas proyek tidak dapat dimulai hingga seluruh aktivitas pendahulunya ($Pred_i$) telah selesai. Contoh aktivitas dalam proyek meliputi "Persiapan lahan" sebagai aktivitas awal dengan durasi 5 hari dan tidak memiliki aktivitas pendahulu. Aktivitas ini membutuhkan sumber daya manusia berupa 2 operator alat berat dan 4 pekerja lapangan selama 5 hari. Aktivitas berikutnya adalah "Pondasi bangunan" dengan durasi 10 hari, yang dapat dimulai setelah "Persiapan lahan" selesai. Aktivitas ini memerlukan 3 tukang bangunan dan 2 mandor selama 10 hari. Selanjutnya,

"Pemasangan kerangka" memiliki durasi 8 hari dan membutuhkan 4 tukang las serta 2 pekerja pembantu selama 8 hari, dengan ketentuan bahwa aktivitas ini hanya dapat dimulai setelah "Pondasi bangunan" selesai. Sumber daya manusia yang digunakan mencakup berbagai tipe seperti operator alat berat, tukang bangunan, mandor, tukang las, dan pekerja pembantu, dengan durasi kerja yang disesuaikan dengan kebutuhan setiap aktivitas.

Selain mempertimbangkan durasi dan hubungan ketergantungan antar-aktivitas, RCPSP juga memerhatikan penggunaan sumber daya secara efisien. Setiap aktivitas membutuhkan sejumlah sumber daya tertentu ($r_{ik}$) untuk setiap tipe $k$, dan jumlah sumber daya yang digunakan pada satu waktu tidak boleh melebihi kapasitas yang tersedia. Oleh karena itu, salah satu tantangan dalam RCPSP adalah memastikan bahwa sumber daya dialokasikan secara optimal untuk menghindari konflik atau keterlambatan selama proses pelaksanaan proyek.

Penjadwalan dilakukan dengan asumsi bahwa seluruh data terkait proyek telah diketahui sebelumnya, termasuk durasi setiap aktivitas, hubungan ketergantungan, dan jumlah sumber daya yang dibutuhkan. Proyek direncanakan tanpa adanya gangguan eksternal, seperti tenaga kerja yang berhenti atau sakit. Dengan demikian, masalah ini berfokus pada perancangan jadwal yang optimal sehingga seluruh aktivitas proyek dapat diselesaikan sesuai dengan batas waktu yang telah ditentukan, dengan tetap memperhatikan efisiensi penggunaan sumber daya.

## 3.2 Pengumpulan Data

Data yang diperlukan dalam penelitian ini mencakup informasi terkait aktivitas dalam proyek yang meliputi:

1. daftar aktivitas proyek: macam-macam aktivitas yang akan dikerjakan,
2. durasi aktivitas: waktu yang dibutuhkan untuk menyelesaikan setiap aktivitas,
3. ketergantungan antar-aktivitas: hubungan ketergantungan yang menunjukkan urutan pelaksanaan aktivitas,
4. sumber daya: informasi terkait ketersediaan dan alokasi sumber daya untuk setiap aktivitas proyek.

Data dikumpulkan melalui wawancara dengan pihak proyek untuk memahami hubungan ketergantungan antar-aktivitas, analisis dokumen perencanaan proyek

untuk memastikan kelengkapan daftar aktivitas, serta pengamatan langsung pada proyek konstruksi serupa untuk memperoleh informasi terkait sumber daya.

Data yang dikumpulkan dalam tahap ini akan digunakan sebagai inisialisasi awal dalam proses optimasi. Dengan menggunakan data yang akurat dan relevan, diharapkan solusi yang dihasilkan dapat lebih realistis dan sesuai dengan kondisi proyek yang sebenarnya. Inisialisasi ini penting untuk memberikan titik awal yang baik, sehingga proses pencarian solusi yang lebih optimal dapat dilakukan dengan lebih efisien dan menghindari solusi yang tidak *feasible.*

### 3.3 Model Optimasi

Model optimasi penjadwalan proyek ini dibangun berdasarkan asumsi-asumsi berikut:

1. semua aktivitas memiliki durasi yang pasti dan telah diketahui sebelumnya,
2. setiap aktivitas hanya dapat dimulai setelah aktivitas yang bergantung padanya selesai,
3. jumlah dan alokasi tenaga kerja bersifat tetap selama periode penjadwalan, tanpa adanya tenaga kerja yang bergabung atau berhenti,
4. selama periode penjadwalan, semua tenaga kerja dalam keadaan sehat dan tidak mengambil cuti, dan
5. penjadwalan dilakukan untuk seluruh durasi proyek tanpa adanya gangguan dari luar.

Sebelum membangun model optimasi, terlebih dahulu didefinisikan himpunan dan parameter model. Berikut adalah pendefinisian himpunan, parameter, dan variabel keputusan yang akan digunakan pada model optimisasi.

1. Himpunan

   Himpunan yang akan digunakan dalam penjadwalan proyek adalah:

   a. $A$: Himpunan aktivitas untuk proyek yang terdiri dari $n + 2$ aktivitas, $A = \{0,1,2, \dots, n + 1\}$. Aktivitas 0 dn $n + 1$ merupakan aktivitas $dummy$ yang menyatakan ‘proyek mulai’ dan ‘proyek selesai’. Aktivitas $dummy$ tidak memiliki durasi dan tidak membutuhkan sumber daya.

   b. $Pred_i$: himpunan aktivitas pendahulu (*predecessor*) bagi aktivitas $i$.

$$Pred_i = \{j | M_{j,i} = 1\}$$

c. $Suc_i$: himpunan aktivitas setelahnya (*successor*) bagi aktivitas $i$.

$$Suc_i = \{j | M_{i,j} = 1\}$$

d. $d$: himpunan durasi aktivitas.

$d = \{d_i | i \in A\}$ dengan $d_i$ adalah durasi aktivitas $i$ dan $d_0 = d_{n+1} = 0$

e. $K$: himpunan semua tipe sumber daya.

$K = \{1, 2, \dots, K\}$ dengan $K$ menyatakan banyaknya tipe sumber daya.

f. $r$: himpunan kebutuhan sumber daya untuk setiap aktivitas.

$r = \{r_{ik} | i \in A, k \in K\}$ dengan $r_{ik}$ adalah banyaknya sumber daya tipe $k$ yang dibutuhkan oleh aktivitas $i$ dan $r_{0,k} = r_{(n+1),k} = 0$.

g. $R$: himpunan ketersediaan setiap tipe sumber daya.

$R = \{R_k | k \in K\}$ dengan $R_k$ adalah jumlah sumber daya tipe $k$ yang tersedia.

2. Parameter

Parameter-parameter yang digunakan dalam penjadwalan proyek mencakup:

a. $d_i$: durasi waktu yang dibutuhkan untuk menyelesaikan aktivitas $i$, dengan $i \in A$.

b. $R_k$: kapasitas maksimum untuk sumber daya tipe $k$, dengan $k \in K$.

c. $r_{ik}$: jumlah sumber daya tipe $k$ yang dibutuhkan oleh aktivitas $i$, dengan $i \in A$ dan $k \in K$.

d. $A_t$: aktivitas yang berjalan pada waktu ke-$t$.

e. $b$: *ready-time,* yaitu waktu yang diperbolehkan untuk memulai proyek.

f. $c$: *deadline*, yaitu batas waktu yang direncanakan untuk menyelesaikan proyek.

g. $t_{mulai,i}$: waktu mulai aktivitas $i$, dengan $i \in A$.

h. $t_{selesai,i}$: waktu selesai aktivitas $i$, dengan $i \in A$.

3. Variabel Keputusan

Variabel keputusan yang digunakan dalam penjadwalan proyek meliputi:

a. $x_{i,t}$: menunjukkan apakah aktivitas $i$ dijadwalkan pada waktu $t$, dengan $i \in A$ dan $t \in T$.

$$x_{i,t} = \begin{cases} 1, \text{jika aktivitas } i \text{ dijadwalkan pada waktu } t \\ 0, \text{lainnya.} \end{cases}$$

b. $M$: matriks berukuran $(n+2) \times (n+2)$ merepresentasikan *precedence and successor relations.* Untuk setiap $i, j \in A$ berlaku.

$$M_{i,j} = \begin{cases} 1, \text{jika aktivitas } i \text{ adalah pendahulu bagi } j \text{ atau } j \text{ adalah penerus } i \\ 0, \text{lainnya.} \end{cases}$$

4. Kendala

Kendala-kendala yang berlaku dalam penjadwalan proyek antara lain:

a. Hubungan antar aktivitas

Aktivitas yang memiliki ketergantungan (*predecessor*) harus diselesaikan terlebih dahulu sebelum aktivitas berikutnya dimulai. Hal tersebut dapat dituliskan:

$$f_h \leq f_i - d_i, \quad h \in Pred_i;\ i = 1, 2, \dots, n+1.$$

Kendala ini memastikan bahwa aktivitas $i$ hanya dapat selesai jika semua aktivitas pendahulunya ( $h \in Pred_i$) telah diselesaikan. Nilai $f_h$, yang melambangkan waktu penyelesaian aktivitas pendahulu $h$, menunjukkan kapan aktivitas $h$ selesai. Aktivitas $h$ harus diselesaikan sebelum aktivitas $i$ dapat dimulai. Dengan demikian, aktivitas $i$ tidak dapat selesai lebih awal dari total waktu penyelesaian pendahulunya ditambah durasinya sendiri $(f_i - d_i)$.

b. Ketersediaan sumber daya

Jumlah sumber daya yang digunakan oleh semua aktivitas pada waktu $t$ tidak boleh melebihi kapasitas sumber daya tipe $k$.

$$\sum_{i \in A_t} r_{ik} x_{it} \leq R_k, \forall k \in K, ; t \geq 0$$

Kendala ini memastikan bahwa pada setiap waktu $t$, jumlah total sumber daya tipe $k$ yang digunakan oleh semua aktivitas yang sedang berjalan tidak melebihi kapasitas sumber daya yang tersedia $(R_k)$. Dengan demikian, proyek harus dijadwalkan sedemikian rupa agar tidak ada kelebihan penggunaan sumber daya pada waktu tertentu.

c. Waktu mulai proyek

$$f_0 \geq b$$

Kondisi ini menyatakan bahwa proyek tidak boleh dimulai sebelum waktu yang diizinkan, yaitu $b$. Dengan kata lain, waktu penyelesaian

aktivitas awal ($f_0$) harus sama atau lebih besar dari waktu yang diperbolehkan untuk memulai proyek.

5. Fungsi tujuan

   Meminimalkan waktu penyelesaian proyek

$$min\ f_{n+1}$$

   di mana $f_{n+1}$ adalah waktu penyelesaian proyek, yang ditentukan oleh waktu penyelesaian aktivitas terakhir ( $n+1$ ) yaitu aktivitas "proyek selesai". Aktivitas ini merupakan penanda bahwa seluruh aktivitas dalam proyek telah diselesaikan. Oleh karena itu, $f_{n+1}$ merepresentasikan waktu penyelesaian proyek secara keseluruhan.

   Selengkapnya, model RCPSP dituliskan sebagai model optimisasi berikut:

   **Meminimumkan:**

$$f_{n+1}$$

   **terhadap**:

$$f_h \leq f_i - d_i, \quad h \in Pred_i;\ i = 1, 2, \dots, n+1,$$

$$\sum_{i \in A_t} r_{ik} x_{it} \leq R_{k,} \forall k \in K, ; t \geq 0,$$

$$f_0 \geq b.$$

## 3.4 Penyelesaian Model

Bagian ini menjelaskan langkah-langkah penerapan metode CPM dan SA untuk menyelesaikan masalah penjadwalan proyek. CPM digunakan untuk mengidentifikasi jalur kritis proyek, sedangkan SA digunakan untuk mengoptimalkan jadwal dengan mempertimbangkan berbagai kendala sumber daya.

### 3.4.1 *Critical Path Method* (CPM)

CPM adalah metode analisis jaringan yang digunakan untuk menentukan waktu penyelesaian proyek, mengidentifikasi jalur kritis, dan memastikan aktivitas-aktivitas yang memengaruhi durasi total proyek (Nalhadi & Suntana, 2017). Hasil perhitungan CPM ini akan digunakan untuk inisialisasi jadwal tetangga baru dalam algoritma optimasi.

Dengan menggunakan jadwal yang dihasilkan dari CPM sebagai titik awal, proses pencarian solusi dapat dimulai dari kondisi yang valid dan sesuai dengan ketergantungan antar aktivitas. Hal ini membantu memastikan bahwa perubahan jadwal yang dilakukan selama pencarian solusi tetap dalam batasan yang memungkinkan, tanpa melanggar urutan aktivitas yang sudah ditentukan sebelumnya.

1. Identifikasi aktivitas dan ketergantungan

   Tahap pertama dalam CPM adalah mengidentifikasi seluruh aktivitas yang harus dilakukan dalam proyek misalkan survei lokasi, persiapan lahan, *finishing*. Setiap aktivitas memiliki durasi yang harus diketahui. Selain itu, kita juga perlu mengetahui hubungan ketergantungan antar aktivitas (*predecessor* dan *successor*).

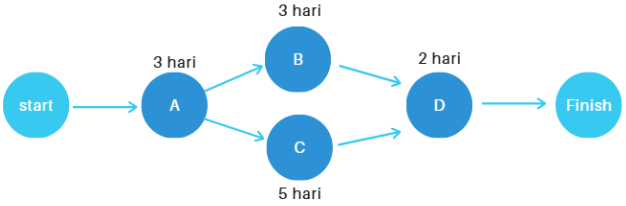
   Contohnya:

   - Aktivitas A harus diselesaikan sebelum aktivitas B dapat dimulai (*predecessor*).
   - Aktivitas C dapat dimulai hanya setelah aktivitas A selesai (*successor*).

   identifikasi ini penting untuk menentukan urutan dan hubungan antar aktivitas, yang selanjutnya akan digunakan dalam perhitungan waktu. Sebagai ilustrasi, berikut adalah contoh sederhana diagram aktivitas dan ketergantungan dalam proyek:

| Kode Aktivitas | Durasi (hari) | *Predecessor* |
|---|---|---|
| A | - | |
| B | 3 | A |
| C | 5 | A |
| D | 2 | B, C |

Dari data tersebut, dapat dibuat diagram jaringan sebagai berikut:




Gambar 3.1 Contoh Diagram Sederhana

Diagram ini menunjukkan bahwa:

- Aktivitas A adalah awal proyek.
- Aktivitas B dan C bergantung pada A.
- Aktivitas D hanya bisa dimulai setelah B dan C selesai.

2. Perhitungan waktu maju (*forward pass*)

Pada tahap ini, akan dihitung waktu mulai paling awal (*early start* - $ES$) dan waktu selesai paling awal *(early finish* - $EF$) untuk setiap aktivitas proyek. Langkah-langkah perhitungan waktu maju:

- Waktu mulai paling awal ($ES$) untuk aktivitas pertama dimulai dari 0 (misalnya, $ES_1 = 0$ untuk aktivitas 1).
- Untuk aktivitas selanjutnya, waktu mulai paling awal dihitung dengan mencari waktu selesai paling awal dari aktivitas pendahulu yang relevan.

Formula menghitung $ES$ ada pada persamaan:

$$ES_i = \begin{cases} 0 & , \ i = 0 \\ \max\{EF_h \mid h \in Pred_i\} & , \ i \neq 0 \end{cases} \quad (3.1)$$

di mana:

$ES_i$: waktu mulai paling awal untuk aktivitas $i$.

$EF_h$: waktu selesai paling awal dari aktivitas pendahulu $h$.

Selanjutnya hitung waktu selesai paling awal ($EF$) yang dihitung dengan menambahkan durasi aktivitas $d_i$ pada waktu mulai paling awal $ES_i$. Formula menghitung $EF$:

$$EF_i = ES_i + d_i \quad (3.2)$$

di mana:

$EF_i$: waktu penyelesaian paling awal untuk aktivitas $i$.

$d_i$: durasi aktivitas $i$.

3. Perhitungan waktu mundur (*backward pass*)

Setelah menghitung waktu maju, langkah berikutnya adalah perhitungan waktu mundur untuk menentukan waktu mulai paling lambat *(late start* - $LS$) dan waktu selesai paling lambat (*late finish* - $LF$).

Langkah-langkah perhitungan waktu mundur:

- Waktu selesai paling lambat ($LF$) untuk aktivitas terakhir (*final project activity*) adalah waktu batas proyek atau nilai yang ditentukan.

- Untuk aktivitas lainnya, $LF_i$ dihitung dengan mencari waktu mulai paling lambat dari aktivitas-aktivitas penerus (*successors*).

Formula untuk menghitung $LF$

$$LF_i = \begin{cases} c & , \ i = n+1 \\ \min\{LS_k \mid k \in Suc_i\}, & i \neq n+1 \end{cases} \quad (3.3)$$

di mana:

$LF_i$: waktu penyelesaian paling akhir untuk kegiatan $i$.

$LS_k$: waktu mulai lambat dari aktivitas penerus $k$.

$c$: *deadline* proyek.

$Suc_i$: suksesor aktivitas $i$.

Selanjutnya menghitung waktu mulai paling lambat ($LS$) yang dihitung dengan mengurangkan durasi aktivitas $d_i$ dari waktu selesai paling lambat $LF_i$. Formula untuk menghitung $LS$:

$$LS_i = LF_i - d_i \quad (3.4)$$

di mana:

$LS_i$: waktu mulai paling akhir untuk kegiatan $i$.

$d_i$: lama waktu dari kegiatan $i$

4. Identifikasi jalur kritis

   Setelah perhitungan maju dan mundur selesai, jalur kritis dapat ditentukan berdasarkan aktivitas-aktivitas yang *slack*-nya (kelonggaran waktu) sama dengan nol. Kelonggaran waktu ($tf$) dihitung dengan rumus:

   $$tf = LF - EF \quad (3.5)$$

   Aktivitas-aktivitas yang ada pada jalur kritis tidak memiliki kelonggaran waktu, yang berarti jika ada penundaan pada aktivitas tersebut, maka durasi proyek akan bertambah.

5. Penentuan durasi proyek dan optimasi

   a. Durasi proyek

      Durasi total proyek adalah durasi yang dibutuhkan untuk menyelesaikan seluruh proyek dari awal hingga akhir. Durasi proyek ditentukan oleh jalur kritis, karena aktivitas yang berada di jalur kritis adalah yang tidak dapat ditunda tanpa menunda keseluruhan proyek.

Durasi proyek dihitung berdasarkan jalur kritis, yaitu jumlah waktu yang dibutuhkan untuk menyelesaikan seluruh aktivitas yang ada di jalur kritis. Durasi proyek tidak dapat lebih cepat dari durasi yang dihitung berdasarkan jalur kritis karena tidak ada kelonggaran waktu di jalur ini.

Contoh: Jika jalur kritis adalah A → C → D dan durasi aktivitas adalah:

- A: 3 hari,
- C: 5 hari,
- D: 2 hari,

Maka, durasi total proyek adalah 3 + 5 + 2 = 10 hari. Proyek tidak bisa selesai lebih cepat dari 10 hari.

b. Optimasi penjadwalan

Setelah menentukan jalur kritis dan durasi proyek, langkah selanjutnya adalah optimasi. Tujuan optimasi adalah untuk meminimalkan durasi proyek atau mengalokasikan sumber daya secara lebih efisien. Metode lain, seperti SA dapat digunakan untuk menemukan solusi yang lebih optimal dalam hal pengalokasian sumber daya atau penjadwalan aktivitas.

3.4.2 *Simulated Annealing* (SA)

Penelitian ini menggunakan SA untuk mengatasi masalah penjadwalan proyek konstruksi, dengan tujuan utama meminimalkan waktu penyelesaian proyek (*makespan*). Algoritma ini mengatur urutan aktivitas dan waktu mulai, memastikan keterkaitan antar aktivitas serta keterbatasan sumber daya tetap terpenuhi. Hasilnya adalah jadwal yang lebih efisien, mencakup waktu mulai dan selesai setiap aktivitas.

a. Inisialisasi parameter input

Pada tahap pertama dalam penerapan Algoritma SA untuk penjadwalan proyek, penetapan parameter input yang tepat sangat penting. Beberapa parameter yang digunakan dalam algoritma ini antara lain (Silitonga & Apdillah, 2017):

- $T_a$ (suhu awal): menentukan fleksibilitas penerimaan solusi pada awal iterasi. Dalam proyek konstruksi, hal ini membantu mengeksplorasi berbagai kemungkinan urutan aktivitas secara luas.
- $T_f$ (suhu akhir): menjadi batas berhentinya iterasi, di mana solusi sudah cukup stabil dan tidak banyak perubahan.
- $\alpha$ (parameter reduksi suhu): mengontrol penurunan suhu tiap iterasi, agar proses berpindah dari eksplorasi ke eksploitasi berjalan bertahap hingga solusi penjadwalan yang efisien tercapai.

b. Inisialisasi solusi awal

Pada tahap awal penjadwalan proyek, solusi awal ditentukan dengan cara menyusun urutan aktivitas yang sudah ada, berdasarkan data yang tersedia. Urutan ini dibuat dengan memperhatikan durasi masing-masing aktivitas dan ketergantungan antar aktivitas dalam proyek. Penjadwalan awal ini bertujuan untuk memberikan gambaran awal mengenai bagaimana proyek akan dijalankan dan sebagai dasar bagi proses optimasi selanjutnya.

Sebagai contoh, misalkan proyek memiliki beberapa aktivitas yang harus diselesaikan, seperti survei lokasi (A) memerlukan waktu 5 hari dan tidak memiliki ketergantungan sebelumnya. Pekerjaan berikutnya, pembersihan dan persiapan lahan (B) memerlukan waktu 3 hari dan bergantung pada penyelesaian survei lokasi (A). Urutan pekerjaan lain juga disesuaikan dengan data *predecessor*, sehingga seluruh jadwal awal memenuhi semua kendala dependensi.

c. Pembangkitan solusi tetangga

Pada langkah ini, bertujuan untuk menghasilkan solusi tetangga dengan memodifikasi solusi yang telah ada. Solusi tetangga adalah solusi yang sedikit berbeda dengan solusi sebelumnya, dengan tujuan mencari kemungkinan penjadwalan yang lebih baik. Pembangkitan solusi tetangga dilakukan dengan mengubah urutan atau waktu mulai aktivitas dalam penjadwalan proyek.

Langkah-langkah pembangkitan solusi tetangga:

1) Pemilihan aktivitas acak

Pada setiap iterasi, satu atau lebih aktivitas yang tidak termasuk dalam jalur kritis dipilih secara acak. Jalur kritis adalah urutan aktivitas yang paling menentukan waktu penyelesaian proyek, sehingga perubahan pada jalur kritis akan langsung mempengaruhi durasi proyek secara keseluruhan.

2) Penggeseran waktu mulai aktivitas

Setelah aktivitas dipilih, waktu mulai salah satu aktivitas tersebut digeser. Aktivitas ini bisa digeser agar terkoordinasi dengan aktivitas lain pada jalur kritis, atau bisa juga digeser terpisah dari aktivitas lainnya, tetap mempertahankan ketergantungan dengan *predecessor*-nya. Penggeseran waktu ini akan mempengaruhi jadwal aktivitas yang lain yang memiliki ketergantungan terhadapnya.

3) Verifikasi ketergantungan (*predecessor*)

Aktivitas yang digeser harus tetap mematuhi ketergantungan *(predecessor*) yang sudah ditetapkan. Misalnya, jika suatu aktivitas B bergantung pada A (contoh, B → A), maka waktu mulai B tidak boleh lebih awal dari waktu selesai A.

d. Penghitungan perubahan biaya

Pada masalah optimisasi penjadwalan proyek, biaya dihitung sebagai nilai pemenuhan fungsi tujuan dari solusi yang terbentuk. Perubahan biaya solusi tetangga dibandingkan dengan solusi awal dapat dihitung menggunakan rumus berikut, misal diketahui waktu selesai proyek terakhir pada solusi awal 65 hari dan waktu selesai proyek terakhir solusi tetangga 57 hari, maka didapat:

$$\delta = f_{n+1} - f_{n+1}{}' = 65 - 57 = 8$$

dengan:

$f_{n+1}$: waktu selesai proyek terakhir pada solusi sebelumnya.

$f_{n+1}{}'$: waktu selesai proyek terakhir solusi selanjutnya.

Jika perubahan biaya ($\delta$) lebih besar dari atau sama dengan nol, maka solusi tetangga diterima sebagai solusi baru. Namun, jika $\delta$ kurang dari nol, perhitungan probabilitas ($P(\delta)$) diperlukan apakah solusi tetangga akan diterima.

$$P(\delta) = exp\left(-\frac{|\delta|}{T_a}\right)$$

di mana:

δ: perubahan biaya antara solusi sebelumnya dan solusi selanjutnya.

$T_a$: suhu awal.

Dalam hal ini karena hasil perhitungan $\delta$ bernilai 8, maka solusi tetangga baru diterima tanpa perlu perhitungan probabilitas. Hal ini menunjukkan bahwa solusi tetangga lebih baik, karena menghasilkan waktu penyelesaian yang lebih cepat yaitu 57 hari dibandingkan dengan 65 hari pada solusi awal.

Jika $\delta < 0$ maka diperlukannya perhitungan probabilitas sebagai contoh pada iterasi pertama suhu adalah $T_1 = 100$, maka probabilitas menerima solusi tetangga dapat dihitung dengan:

$$P(\delta) = exp\left(-\frac{|8|}{100}\right) \approx 0.923$$

Setelah menghitung probabilitas, akan dibandingkannya dengan nilai acak yang dihasilkan (misalnya, $r \in [0,1]$).

- Jika $P(\delta) \geq r$, maka solusi tetangga diterima.
- Jika $P(\delta) < r$, maka solusi tetangga ditolak, dan langkah ini akan diulang untuk mencari solusi tetangga lainnya.

Misalnya, nilai acak yang dihasilkan $r = 0.85$, karena $P(\delta) \geq r$ maka solusi tetangga akan diterima. Setelah langkah penerimaan atau penolakan solusi, suhu akan disusutkan untuk memperkecil kemungkinan menerima solusi yang lebih buruk.

e. Proses pendinginan

Proses pendinginan berperan penting dalam algoritma SA. Tujuan dari proses ini adalah untuk mengontrol seberapa besar kemungkinan menerima solusi yang lebih buruk (lebih lama atau lebih mahal) saat algoritma mencari solusi optimal.

- Pada awalnya, suhu (T) cukup tinggi. Hal ini memungkinkan algoritma menerima solusi yang lebih buruk untuk menghindari terjebak pada solusi lokal.

- Seiring berjalannya waktu, suhu (T) secara bertahap akan berkurang. Seiring penurunan suhu, kemungkinan untuk menerima solusi yang lebih buruk juga semakin kecil, sehingga algoritma akan semakin fokus pada menemukan solusi yang lebih baik dan optimal.

  Proses pendinginan dapat digambarkan dengan rumus:

  $$T_{a+1} = \alpha . T_a$$

  di mana:

  $T_a$: suhu pada iterasi ke-$a$

  $\alpha$: faktor pendinginan

  contoh: Jika suhu pada iterasi pertama $T_1 = 100$ dan faktor pendinginan $\alpha = 0{,}9$, maka suhu pada iterasi berikutnya $T_2 = 90$, dan suhu terus berkurang seiring iterasi.

Proses pendinginan dalam SA pada RCPSP akan memastikan bahwa meskipun solusi awal yang lebih buruk dapat diterima pada awal pencarian, ketika waktu berjalan, algoritma semakin fokus untuk mencari solusi yang lebih baik, yaitu solusi dengan waktu penyelesaian lebih singkat, yang memenuhi kendala sumber daya, dan mempertimbangkan ketergantungan antar aktivitas.

f. Penghentian algoritma

Proses penghentian algoritma SA bergantung dilakukan berdasarkan beberapa kondisi, salah satunya adalah ketika suhu ($T$) mendekati nol. Ketika suhu sangat rendah, perubahan lebih lanjut dalam solusi sangat kecil, dan algoritma dapat dianggap telah menemukan solusi yang hampir optimal. Algoritma dapat dihentikan ketika $T$ sangat kecil, misalnya $T \leq T_{min}$, di mana $T_{min}$ adalah suhu batas yang telah ditentukan sebelumnya.

## 3.5 Contoh Kasus dan Penyelesaian

Bagian ini menjelaskan penerapan metode pada sebuah studi kasus untuk menyelesaikan masalah penjadwalan proyek. Studi kasus ini melibatkan data aktivitas, durasi, dan ketergantungan antar aktivitas. Hasil analisis digunakan untuk menunjukkan langkah-langkah penyelesaian dan efektivitas metode dalam memecahkan masalah penjadwalan.

3.5.1 Data

Data yang digunakan sebagai contoh kasus adalah proyek konstruksi sederhana dengan rincian aktivitas, sumber daya yang dibutuhkan, dan durasi pengerjaan. Tabel 3.1 memberikan detail informasi penjadwalan proyek:

Tabel 3.1 Contoh Data Penjadwalan Proyek Konstruksi

| Kode | Deskripsi Pekerjaan | $r_1$ | $r_2$ | $r_3$ | $r_4$ | $d_i$ | $Pred_i$ |
|---|---|---|---|---|---|---|---|
| A | Survei lokasi | 2 | 1 | 4 | 0 | 5 | - |
| B | Pembersihan dan persiapan lahan | 1 | 1 | 6 | 0 | 3 | A |
| C | Penggalian dan pemasangan pondasi | 2 | 2 | 8 | 0 | 7 | B |
| D | Pembuatan struktur lantai dasar | 1 | 1 | 5 | 1 | 10 | C |
| E | Pemasangan dinding dan kolom | 1 | 1 | 7 | 1 | 8 | D |
| F | Pemasangan rangka atap | 2 | 1 | 5 | 1 | 5 | E |
| G | Pemasangan instalasi listrik dan pipa air | 1 | 1 | 6 | 0 | 8 | D |
| H | Plesteran dan finishing dinding | 1 | 1 | 4 | 1 | 7 | E |
| I | Pengecatan dinding | 1 | 1 | 4 | 1 | 7 | H |
| J | Pembersihan akhir dan serah terima | 1 | 1 | 3 | 0 | 5 | F, G, I |

Tipe sumber daya yang digunakan terdiri atas pekerja logistik ($r_1$) dengan kapasitas maksimum sebanyak $R_1$ = 3. Manajer proyek ($r_2$) dengan kapasitas maksimum $R_2$ = 2. Pekerja lapangan ($r_3$) dengan kapasitas maksimum $R_3$ = 10. Serta tenaga ahli keselamatan kerja ($r_4$) dengan kapasitas maksimum $R_4$ = 2. Selain itu, $d_i$ menunjukkan durasi pengerjaan setiap aktivitas dalam satuan waktu, dan $Pred_i$ merupakan aktivitas pendahulu yang harus diselesaikan terlebih dahulu sebelum aktivitas tersebut dapat dimulai.

3.5.2 *Critical Path Method* (CPM)

Data yang disajikan dalam Tabel 3.1 akan diimplementasikan menggunakan metode CPM. CPM bertujuan untuk menentukan jalur kritis proyek, yaitu rangkaian aktivitas dengan durasi total terpanjang yang tidak dapat ditunda tanpa memperpanjang waktu penyelesaian proyek.

a. Perhitungan maju

1. Pekerjaan dengan kode A:

$$ES_A = 0 \text{ (karena tidak adak pendahulu) dan}$$

$$EF_A = ES_A + D_A = 0 + 5 = 5.$$

2. Pekerjaan dengan kode B:

$$ES_B = EF_A = 5 \text{ dan}$$

$$EF_B = ES_B + D_B = 5 + 3 = 8.$$

3. Pekerjaan dengan kode C:

$$ES_C = EF_B = 8 \text{ dan}$$
$$EF_C = ES_C + D_C = 8 + 7 = 15.$$

4. Pekerjaan dengan kode D:

$$ES_D = EF_C = 15 \text{ dan}$$
$$EF_D = ES_D + D_D = 15 + 10 = 25.$$

5. Pekerjaan dengan kode E:

$$ES_E = EF_D = 25 \text{ dan}$$
$$EF_E = ES_E + D_E = 25 + 8 = 33.$$

6. Pekerjaan dengan kode F:

$$ES_F = EF_E = 33 \text{ dan}$$
$$EF_F = ES_F + D_F = 33 + 5 = 38.$$

7. Pekerjaan dengan kode G:

$$ES_G = EF_D = 25 \text{ dan}$$
$$EF_G = ES_G + D_G = 25 + 8 = 33.$$

8. Pekerjaan dengan kode H:

$$ES_H = EF_E = 33 \text{ dan}$$
$$EF_H = ES_H + D_H = 33 + 7 = 40.$$

9. Pekerjaan dengan kode I:

$$ES_I = EF_H = 40 \text{ dan}$$
$$EF_I = ES_I + D_I = 40 + 7 = 47.$$

10. Pekerjaan dengan kode J:

$$ES_J = \max(EF_F, EF_G, EF_I) = \max(38,33,47) = 47 \text{ dan}$$
$$EF_J = ES_J + D_J = 47 + 5 = 52.$$

b. Perhitungan mundur

1. Pekerjaan dengan kode J:

$$LF_J = EF_J = 52 \text{ dan}$$
$$LS_J = LF_J - D_J = 52 - 5 = 47.$$

2. Pekerjaan dengan kode I:

$$LF_I = LS_J = 47 \text{ dan}$$

$$LS_I = LF_I - D_I = 47 - 7 = 40.$$

3. Pekerjaan dengan kode H:

$$LF_H = LS_I = 40 \text{ dan}$$
$$LS_H = LF_H - D_H = 40 - 7 = 33.$$

4. Pekerjaan dengan kode G:

$$LF_G = LS_J = 47 \text{ dan}$$
$$LS_G = LF_G - D_G = 47 - 8 = 39.$$

5. Pekerjaan dengan kode F:

$$LF_J = LS_J = 47 \text{ dan}$$
$$LS_F = LF_F - D_F = 47 - 5 = 42.$$

6. Pekerjaan dengan kode E:

$$LF_E = \min(LS_F, LS_H) = \min(42,33) = 33 \text{ dan}$$
$$LS_E = LF_E - D_E = 33 - 8 = 25.$$

7. Pekerjaan dengan kode D:

$$LF_D = \min(LS\_E, LS\_G) = \min(25,39) = 25 \text{ dan}$$
$$LS_D = LF_D - D_D = 25 - 10 = 15.$$

8. Pekerjaan dengan kode C:

$$LF_C = LS_D = 8 \text{ dan}$$
$$LS_C = LF_C - D_C = 15 - 7 = 8.$$

9. Pekerjaan dengan kode B:

$$LF_B = LS_C = 8 \text{ dan}$$
$$LS_B = LF_B - D_B = 8 - 3 = 5.$$

10. Pekerjaan dengan kode A:

$$LF_A = LS_B = 5 \text{ dan}$$
$$LS_A = LF_A - D_A = 5 - 5 = 0.$$

Ringkasan hasil analisis jadwal proyek konstruksi berdasarkan metode CPM disajikan pada Tabel 3.2. Tabel tersebut memuat informasi mengenai kode pekerjaan, deskripsi pekerjaan, hubungan antar aktivitas (*predecessor*), serta nilai *early start* ($ES$), *early finish* ($EF$), *late start* ($LS$), dan *late finish* ($LF$) untuk masing-masing aktivitas.

Tabel 3.2 Penjadwalan Proyek berdasarkan Metode CPM

| Kode | Deskripsi Pekerjaan | $Pred_i$ | $ES$ | $EF$ | $LS$ | $LF$ |
|---|---|---|---|---|---|---|
| A | Survei lokasi | - | 0 | 5 | 0 | 5 |
| B | Pembersihan dan persiapan lahan | A | 5 | 8 | 5 | 8 |
| C | Penggalian dan pemasangan pondasi | B | 8 | 15 | 8 | 15 |
| D | Pembuatan struktur lantai dasar | C | 15 | 25 | 15 | 25 |
| E | Pemasangan dinding dan kolom | D | 25 | 33 | 25 | 33 |
| F | Pemasangan rangka atap | E | 33 | 38 | 42 | 47 |
| G | Pemasangan instalasi listrik dan pipa air | D | 25 | 33 | 39 | 47 |
| H | Plesteran dan finishing dinding | E | 33 | 40 | 33 | 40 |
| I | Pengecatan dinding | H | 40 | 47 | 40 | 47 |
| J | Pembersihan akhir dan serah terima | F, G, I | 47 | 52 | 47 | 52 |

Setelah melakukan perhitungan maju dan mundur, dilakukan pula perhitungan *float* (*slack*). Suatu kegiatan dikategorikan sebagai kegiatan kritis apabila memiliki nilai total *float* ($tf$) sama dengan nol ($tf = 0$), atau ketika nilai perhitungan maju ($ES$ dan $EF$) sama dengan nilai perhitungan mundur ($LS$ dan $LF$), di mana $tf$ dihitung menggunakan formula $tf = LS - ES$. Tabel 3.3 menyajikan hasil perhitungan parameter jadwal, termasuk total float untuk setiap aktivitas proyek.

Tabel 3.3 Hasil Perhitungan *Total Float* ($\boldsymbol{tf}$) pada Jadwal Proyek

| Kode | Deskripsi Pekerjaan | $Pred_i$ | $ES$ | $EF$ | $LS$ | $LF$ | $tf$ |
|---|---|---|---|---|---|---|---|
| A | Survei lokasi | - | 0 | 5 | 0 | 5 | 0 |
| B | Pembersihan dan persiapan lahan | A | 5 | 8 | 5 | 8 | 0 |
| C | Penggalian dan pemasangan pondasi | B | 8 | 15 | 8 | 15 | 0 |
| D | Pembuatan struktur lantai dasar | C | 15 | 25 | 15 | 25 | 0 |
| E | Pemasangan dinding dan kolom | D | 25 | 33 | 25 | 33 | 0 |
| F | Pemasangan rangka atap | E | 33 | 38 | 42 | 47 | 9 |
| G | Pemasangan instalasi listrik dan pipa air | D | 25 | 33 | 39 | 47 | 14 |
| H | Plesteran dan finishing dinding | E | 33 | 40 | 33 | 40 | 0 |
| I | Pengecatan dinding | H | 40 | 47 | 40 | 47 | 0 |
| J | Pembersihan akhir dan serah terima | F, G, I | 47 | 52 | 47 | 52 | 0 |

Kegiatan yang memenuhi syarat $tf = 0$ membentuk jalur kritis proyek, yaitu: A → B → C → D → E → H → I → J. Aktivitas-aktivitas ini harus diselesaikan tepat waktu agar tidak terjadi keterlambatan dalam penyelesaian proyek. Sebaliknya, aktivitas yang memiliki nilai $tf > 0$ termasuk dalam kategori

kegiatan non-kritis, yang memiliki fleksibilitas waktu tertentu tanpa memengaruhi jadwal keseluruhan proyek.

3.5.3 *Simulated Annealing* (SA)

SA digunakan untuk mengoptimalkan jadwal proyek dengan mencari solusi terbaik secara bertahap. Metode ini mempertimbangkan durasi dan ketergantungan aktivitas untuk menghasilkan jadwal yang efisien dan meminimalkan keterlambatan.

a. Inisialisasi parameter input

Proses SA diawali dengan inisialisasi parameter input yang diperlukan untuk algoritma. Parameter-parameter tersebut meliputi suhu awal ($T_a$), suhu akhir ($T_f$), dan faktor reduksi suhu ($\alpha$). Nilai-nilai parameter beserta rentang umumnya ditunjukkan pada Tabel 3.4.

Tabel 3.4 Parameter Input dan Rentang Nilai pada SA

| Parameter Input | Rentang Nilai | Nilai |
|---|---|---|
| Suhu Awal ($T_a$) | > 0 | 100 |
| Suhu Akhir ($T_f$ ) | $0 < T_f < T_a$ | 1 |
| Faktor Reduksi Suhu ($\alpha$) | 0 – 1 | 0 – 1 |

Menurut Muhaddad, (2014) Nilai $T_a$ harus lebih dari 0, dan dalam penelitian ini ditetapkan sebesar 100. Suhu akhir ditentukan sebesar 1 sebagai batas berhentinya iterasi, sesuai dengan ketentuan bahwa $0 < T_f < T_a$ (Panggabean, 2004). Faktor reduksi suhu (α) digunakan untuk menurunkan suhu secara bertahap pada setiap iterasi dengan nilai antara 0 dan 1 (Muhaddad, 2014).

b. Inisialisasi solusi awal

Inisialisasi nilai awal pada proses ini dilakukan dengan menggunakan jadwal proyek yang telah ditentukan sebelumnya. Secara teori, pemilihan solusi awal tidak memengaruhi kualitas solusi akhir, karena solusi cenderung konvergen secara keseluruhan. Namun, penerapan metode heuristik yang tepat dalam pembentukan solusi awal dapat mempercepat waktu komputasi untuk mencapai solusi akhir. Dalam penelitian ini, solusi awal didasarkan pada jadwal penjadwalan proyek sebagaimana ditampilkan pada Tabel 3.5. Jadwal ini digunakan sebagai dasar untuk memulai proses algoritma.

Tabel 3.5 Jadwal Proyek Sebagai Solusi Awal

| Kode | Deskripsi Pekerjaan | $r_1$ | $r_2$ | $r_3$ | $r_4$ | $d_i$ | $t_{mulai,i}$ | $t_{selesai,i}$ |
|---|---|---|---|---|---|---|---|---|
| A | Survei lokasi | 2 | 1 | 4 | 0 | 5 | 0 | 5 |
| B | Pembersihan dan persiapan lahan | 1 | 1 | 6 | 0 | 3 | 5 | 8 |
| C | Penggalian dan pemasangan pondasi | 2 | 2 | 8 | 0 | 7 | 8 | 15 |
| D | Pembuatan struktur lantai dasar | 1 | 1 | 5 | 1 | 10 | 15 | 25 |
| E | Pemasangan dinding dan kolom | 1 | 1 | 7 | 1 | 8 | 25 | 33 |
| F | Pemasangan rangka atap | 1 | 1 | 5 | 1 | 5 | 33 | 38 |
| G | Pemasangan instalasi listrik dan pipa air | 1 | 1 | 6 | 0 | 8 | 38 | 46 |
| H | Plesteran dan finishing dinding | 1 | 1 | 4 | 1 | 7 | 46 | 53 |
| I | Pengecatan dinding | 1 | 1 | 4 | 1 | 7 | 53 | 60 |
| J | Pembersihan akhir dan serah terima | 1 | 1 | 3 | 0 | 5 | 60 | 65 |

Berdasarkan jadwal tersebut, waktu penyelesaian proyek diperoleh sebesar 65 hari, dengan nilai maksimum solusi awal penyelesaian aktivitas ($f_{n+1}$) sama dengan 65 hari.

c. Pembangkitan solusi tetangga

Pembangkitan solusi tetangga merupakan langkah penting dalam iterasi algoritma untuk mencari solusi optimal. Dalam penelitian ini, solusi tetangga dihasilkan menggunakan Algoritma *Simple Searching Neighbourhood*, yang diadaptasi dari Aycan dan Ayav (2009). Solusi tetangga dibuat dengan menggeser waktu mulai aktivitas tertentu yang tidak termasuk dalam jalur kritis, yaitu jalur A → B → C → D → E → H → I → J. Aktivitas non-kritis (F dan G) dipilih secara acak untuk digeser sesuai dengan ketentuan *predecessor* dan batasan sumber daya.

Tabel 3.6 Relasi Aktivitas berdasarkan *Predecessor*

| Kode | $Pred_i$ |
|---|---|
| A | - |
| B | A |
| C | B |
| D | C |
| E | D |
| F | E |
| G | D |
| H | E |
| I | H |
| J | F, G, I |

Tabel 3.7 Contoh Solusi Tetangga Awal dengan Pelanggaran Kendala

| Kode | $d_i$ | $t_{mulai,i}$ | $t_{selesai,i}$ | $r_1$ | $r_2$ | $r_3$ | $r_4$ |
|---|---|---|---|---|---|---|---|
| A | 5 | 0 | 5 | 2 | 1 | 4 | 0 |
| B | 3 | 5 | 8 | 1 | 1 | 6 | 0 |
| C | 7 | 8 | 15 | 2 | 2 | 8 | 0 |
| D | 10 | 15 | 25 | 1 | 1 | 5 | 1 |
| E | 8 | 25 | 33 | 1 | 1 | 7 | 1 |
| G | 8 | 25 | 33 | 1 | 1 | 6 | 0 |
| F | 5 | 33 | 38 | 1 | 1 | 5 | 1 |
| H | 7 | 38 | 45 | 1 | 1 | 4 | 1 |
| I | 7 | 45 | 52 | 1 | 1 | 4 | 1 |
| J | 5 | 52 | 57 | 1 | 1 | 3 | 0 |

Setelah berhasil menemukan solusi tetangga, dilakukan verifikasi terhadap pemenuhan kendala. Jika terdapat pelanggaran kendala, solusi tetangga sebelumnya diabaikan dan pencarian solusi tetangga diulang hingga ditemukan solusi tetangga yang memenuhi syarat. Pada Tabel 3.7 melanggar kendala yaitu pada hari ke-25 hingga hari ke-33 tipe sumber daya 3 melebihi batas sumber daya yaitu 10, maka dilakukan pencarian tetangga ulang.

Tabel 3.8 Solusi Tetangga yang Memenuhi Kendala

| Kode | $d_i$ | $t_{mulai,i}$ | $t_{selesai,i}$ | $r_1$ | $r_2$ | $r_3$ | $r_4$ |
|---|---|---|---|---|---|---|---|
| A | 5 | 0 | 5 | 2 | 1 | 4 | 0 |
| B | 3 | 5 | 8 | 1 | 1 | 6 | 0 |
| C | 7 | 8 | 15 | 2 | 2 | 8 | 0 |
| D | 10 | 15 | 25 | 1 | 1 | 5 | 1 |
| E | 8 | 25 | 33 | 1 | 1 | 7 | 1 |
| G | 8 | 33 | 41 | 1 | 1 | 6 | 0 |
| H | 7 | 41 | 48 | 1 | 1 | 4 | 1 |
| I | 7 | 48 | 55 | 1 | 1 | 4 | 1 |
| F | 5 | 48 | 53 | 1 | 1 | 5 | 1 |
| J | 5 | 55 | 60 | 1 | 1 | 3 | 0 |

Setelah pembangkitan solusi tetangga, dilakukan verifikasi untuk memastikan solusi memenuhi semua kendala yang telah ditentukan. Jika ditemukan pelanggaran, seperti pada Tabel 3.7 yang menunjukkan pelanggaran batas sumber daya pada tipe sumber daya 3, solusi tersebut diabaikan dan dilakukan pembangkitan solusi tetangga ulang. Hasil dari proses ini disajikan pada Tabel 3.8, di mana solusi tetangga yang memenuhi kendala digunakan untuk evaluasi fungsi tujuan berikutnya. Pada Tabel 3.8, waktu selesai proyek pada solusi tetangga $f_{n+1}$ tercatat 60 hari.

d. Penghitungan perubahan biaya

Perubahan biaya dihitung dengan membandingkan waktu selesai proyek terakhir pada solusi awal dengan waktu selesai proyek terakhir pada solusi tetangga. Perubahan ini dihitung menggunakan rumus:

$$\delta = f_{n+1} - f_{n+1}'$$

Diketahui bahwa waktu selesai proyek terakhir pada solusi awal $f_{n+1}$ berjumlah 65 hari dan waktu selesai proyek terakhir solusi tetangga $f_{n+1}'$ berjumlah 60 hari maka, perubahan biaya ($\delta$) dihitung sebagai berikut:

$$\delta = f_{n+1} - f'_{n+1} = 65 - 60 = 5$$

karena $\delta \geq 0$ solusi tetangga diterima sebagai solusi baru karena menghasilkan penyelesaian proyek yang lebih cepat atau setara.

e. Proses pendinginan

Pada proses ini suhu awal $T_1 = 100$. Lalu untuk iterasi selanjutnya atau iterasi kedua, diatur dengan faktor reduksi suhu $\alpha$, yang berkisar antara 0 dan 1. Misalnya, jika $\alpha = 0.9$, suhu pada iterasi kedua menjadi:

$$T_2 = T_1 \times \alpha = 100 \times 0{,}9 = 90$$

Suhu ini kemudian digunakan untuk iterasi berikutnya. Proses ini akan berlanjut dengan suhu yang semakin kecil, diikuti dengan perhitungan probabilitas penerimaan solusi tetangga dan pembaruan suhu hingga suhu mencapai suhu akhir $T_f$. Misalnya, jika $T_f = 1$, maka setelah beberapa iterasi suhu akan terus menurun hingga mencapai 1.

f. Penghentian algoritma

Algoritma SA akan berhenti ketika suhu mencapai nilai akhir yang telah ditentukan, yaitu $T_f$, atau ketika jumlah iterasi maksimum tercapai. Nilai $T_f$ yang ditentukan pada kasus ini adalah 1, saat nilai $T_f$ mencapai 1 maka algoritma dihentikan.

g. Hasil sementara

Proses optimasi penjadwalan proyek dengan SA berhasil mengurangi waktu penyelesaian proyek dari 65 hari menjadi 60 hari. Solusi tetangga yang memenuhi kendala sumber daya diterima karena memberikan hasil yang lebih cepat ($\delta = 5$).

Pada iterasi kedua, algoritma dimulai dengan inisialisasi solusi tetangga baru. Langkah ini bertujuan untuk mencari solusi alternatif yang tetap memenuhi kendala sumber daya sambil memperbaiki waktu penyelesaian proyek. Setelah solusi dihasilkan, probabilitas penerimaannya dihitung berdasarkan suhu yang telah diperbarui, memastikan eksplorasi solusi tetap berjalan. Algoritma SA akan berhenti ketika suhu mencapai nilai akhir ($T_f = 1$) atau iterasi maksimum tercapai, memastikan proses berjalan efisien dan mendekati solusi optimal.