Program Evaluation and Review Technique (PERT) Analysis to Predict Completion Time and Project Risk Using Discrete Event System Simulation Method

I Gusti Agung Anom Yudistira[1*], Rinda Nariswari[2], Samsul Arifin[3], Abdul Azis Abdillah[4], Puguh Wahyu Prasetyo[5], Nanang Susyanto[6]
[1,2]Statistics Department, School of Computer Science, Bina Nusantara University
Jakarta, Indonesia 11480
[3]Data Science, Faculty of Engineering and Design, Institut Teknologi Sains Bandung
Bekasi, Indonesia 17530
[4]Mechanical Engineering Department, Politeknik Negeri Jakarta
Depok, Indonesia 16425
[5]Mathematics Education Department, Faculty of Teacher Training and Education,
Universitas Ahmad Dahlan
Yogyakarta, Indonesia 55191
[6]Mathematics Department, Faculty of Mathematics and Natural Sciences, Universitas Gadjah Mada
Yogyakarta, Indonesia 55281
Email: [1]i.yudistira@binus.ac.id, [2]rinda.nariswari@binus.ac.id, [3]samsul.arifin@itsb.ac.id,
[4]abdul.azis.a@mesin.pnj.ac.id, [5]puguh.prasetyo@pmat.uad.ac.id, [6]nanang_susyanto@ugm.ac.id

***Abstract*—The prediction of project completion time, which is important in project management, is only based on an estimate of three numbers, namely the fastest, slowest, and presumably time. The common practice of applying normal distribution through Monte Carlo simulation in Program Evaluation and Review Technique (PERT) research often fails to accurately represent project activity durations, leading to potentially biased project completion prediction. Based on these problems, a different method is proposed, namely, Discrete Event Simulation (DES). The research aims to evaluate the effectiveness of the simmer package in R in conducting PERT analysis. Specifically, there are three objectives in the research: 1) develop a simulation model to predict how long a project will take and find the critical path, 2) create an R script to simulate discrete events on a PERT network, and 3) explore the simulation output using the simmer package in the form of summary statistics and estimation of project risk. Then, a library research with a descriptive and exploratory method is used for data collection. The hypothetical network is used to obtain the numerical results, which provide the predicted value of the project completion, the critical path, and the risk level. Simulation, including 100 replications, results in a predicted project completion time and a standard deviation of 20.7 and 2.2 weeks, respectively. The DES method has been proven highly effective in predicting the completion time of a project described by the PERT network. In addition, it offers increased flexibility.**

***Index Terms*—Program Evaluation and Review Technique (PERT), Completion Time, Project Risk, Discrete Event System Simulation**

Received: May 05, 2022; received in revised form: Dec. 23, 2022; accepted: Dec. 23, 2022; available online: April 29, 2024.
*Corresponding Author

I. Introduction

SIMULATION is often used to explore the complexities of many real-world systems due to the adoption of exorbitant analytical models [1–3]. This problem-solving method predates the invention of computers. However, recent developments have led to the linking of similar tools and methods. Simulation is also the process of designing the model of a real system and conducting experiments to understand the behavior of the system [4, 5]. The simulation process is also used to evaluate various strategies in the limits imposed by a set of criteria for operational purposes [1, 3]. Several forms of simulation are used, depending on the nature of the system under investigation. A standard model

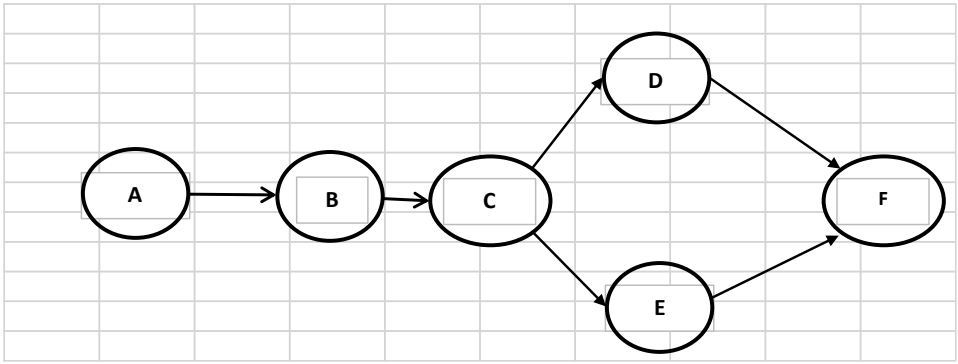

Fig. 1. Example of a simple Program Evaluation and Review Technique (PERT) network.

taxonomy divides simulation issues into three categories: deterministic vs. stochastic, static vs. dynamic depending on the requirement of a temporal component, and continuous vs. discrete based on how the system changes [1, 2]. For example, the Monte Carlo method is a well-known static stochastic simulation method [1, 6, 7].

Another typical example is Discrete Event Simulation (DES), which is designed to model stochastic, dynamic, and discretely changing systems. DES differs from continuous simulation methods, which rely on smoothly evolving equational models [8]. In DES, state transitions occurred abruptly at specified moments in simulated time. Simmer is a DES package for R that facilitates high-level process-oriented modeling, similar to the functionality offered by other modern simulators [9–11]. This package adopts the concept of trajectory, representing a shared path in a simulation model for entities of the same type. In addition, the trajectory is a set of standardized acts that characterize the lifecycle of similar processes. This design pattern is easy to implement and uses the chaining or piping process of the magrittr package [2, 8].

Program Evaluation and Review Technique (PERT) is a method used for planning and controlling non-repetitive projects. The projects include topics that have not been conducted and will not be carried out again in the same manner in the future. The main aim of PERT is to reduce delays, production disruptions, and conflicts while efficiently coordinating and synchronizing various project components to accelerate completion. Graphically, PERT is represented as a network consisting of events and activities. These events and activities are depicted as circles and arrows, respectively. Figure 1 is an example of a network comprising four events (cycles) and activities [4, 12].

PERT is a method to predict the time required to complete a project with probabilistic elements. It adopts a three-digit estimation method, considering the fastest, slowest, and presumably time [12]. In practice, the use of a normal distribution assumption for probabilistic time often leads to biased predictions [13]. The research proposes a new method, namely using DES based on the three estimated time numbers, known as the triangular distribution. Constructing a PERT network requires comprehensive information about all significant project activities or tasks. Accurate mapping of preceding and succeeding activities for each task is crucial. Furthermore, the completion time of an activity can be obtained through a brainstorming process, typically including the fastest, longest, and presumably time. The development of PERT is based on six steps [14, 15], as follows:

1) Define the project and identify all significant activities or tasks.
2) Establish connections between these activities, determining which ones should precede or follow the others.
3) Describe the network that connects all activities.
4) Assign estimated time and cost to each activity.
5) Calculate the critical path, the longest time path through the network.
6) Use the network to help with project planning, scheduling, and control.

The research focuses on the fifth step, particularly in predicting the project or program completion time. Therefore, the research objectives are formulated based on previous studies [16, 17]. It aims to 1) develop a simulation model capable of predicting project completion time and identifying the critical path, 2) develop an R script that simulates discrete events on the PERT network, and 3) explore simulation output using simmer package, both in the form of summary statistics and estimates of project risk. In addition, combining the statistical and graphical analysis of R with the simmer package offers interesting possibilities [18].

The synergy between simmer and other R packages is also analyzed. The main competitors of simmer are SimPy and SimJulia, developed for Python and Julia, respectively [19, 20].

II. Research Method

The research adopts descriptive, exploratory, and library methods [21–23]. The following steps are applied:

1) Assess the literature thoroughly to understand the fundamental principles of PERT. R functions are implemented using R and simmer package to conduct a simulation.
2) Review the literature thoroughly to understand the environmental structure of the simmer class completely.
3) Collect data using generated information based on the PERT network problem for a hypothetical project.
4) Develop a flowchart showing the PERT network of a hypothetical project.
5) Verify the accuracy and reliability of the model developed in the fourth step.
6) Develop R scripts in stages, starting with the simplest form and gradually to the complex one.
7) Run simulations multiple times to predict the completion time of the project and experiment with different scenarios.
8) Analyze simulation output, which includes:
   a) Evaluate the estimated duration of project completion. The research repeats the simulation 100 times because, from the observations, the values obtained are relatively stable. Simulation output from these iterations provides both the predicted value for the project completion time and an estimate for the confidence interval.
   b) Conduct a critical path analysis, which includes identifying available pathways and determining the path with the longest duration achieved through simulation.
   c) Conduct a project risk analysis, specifically assessing the risk of project completion delays. However, through simulation calculations, probabilities of project delays will be determined for various promised completion times.

The research focuses on predicting the project completion time and identifying the critical path in the developed network. Based on predefined lower, upper, and mode values, the hypothetical network has activity durations following a triangular distribution. Therefore, the research can be referenced in more realistic projects.

Next, the PERT network used is hypothetical, reflecting the focus on assessing the effectiveness of discrete event system simulation in project management, particularly in predicting the completion time. The success of the research leads to testing the method on the PERT network with real problems. The research focuses on the time resource and fails to discuss the cost aspect of each activity. In addition, the cost is directly proportional to the amount of time needed to complete an activity. It focuses on the significance of considering time as a crucial resource in project management [12, 13].

III. Results and Discussion

PERT network consists of two essential elements: status, depicted as circles, and activity, represented by arrows marked with specific numbers, respectively [14]. The network provided, representing a project, serves as the basis for discussion, with the completion time predicted using the DES method. The network in Fig. 2 shows a project with nine activities and six states or stages, denoted by circles labeled A to F. Each activity is numbered 1 to 9 in the box [4].

A: The project starts, as well as activities 1, 2, and 3.

B: Activity 1 is completed, triggering the start of 4.

C: Activities 2 and 5 are completed, and 6 and 7 are initiated simultaneously.

D: Activities 3 and 7 are completed, prompting the start of 6 and 9.

E: Activity 4 is completed, leading to the start of 8.

F: Completion of activities 6, 8, and 9, ending the project.

The completion time for each activity follows a triangular distribution with parameters a, b, and m denoted by triangular (a, b, m) [20]. The time units are measured in weeks, with a, b, and m denoting the fastest, longest, and presumably time or mode, respectively. Table I shows the estimation parameters of the triangular distribution for nine activities.

The prediction of project completion time is conducted through the DES method using a simmer package in R. R coding is divided into four parts, starting with defining the duration for each activity. This initialization process is structured as a user-generated function. Three essential packages, namely simmer, EnvStats, and parallel package, are used for simulation, and the `rtri` function is used for triangular distribution and iterating over simulation runs.

In the context of DES, the PERT network shown in Fig. 2 is viewed as an entity trajectory, initiating each

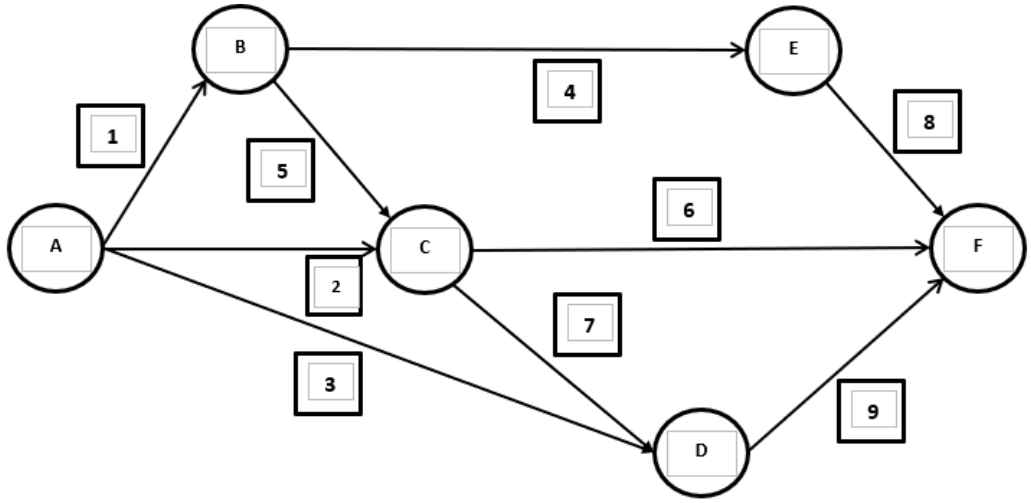

Fig. 2. Program Evaluation and Review Technique (PERT) network for a hypothetical project.

TABLE I
ACTIVITY NAME AND PROCESSING TIME WITH TRIANGULAR DISTRIBUTION (a, b, AND m).

Activity	Processing time
Activity 1	Triangular (1, 5, 3)
Activity 2	Triangular (3, 9, 6)
Activity 3	Triangular (10, 19, 13)
Activity 4	Triangular (3, 12, 9)
Activity 5	Triangular (1, 8, 3)
Activity 6	Triangular (8, 16, 9)
Activity 7	Triangular (4, 13, 7)
Activity 8	Triangular (3, 9, 6)
Activity 9	Triangular (1, 8, 3)

process. The steps applied in writing the R script are as follows:

1) The initialization process includes two main steps: a) producing a simulation environment using a simmer function named `PERT2` and b) defining the completion duration for activity `i` (`duration_i`) to obtain the value of the triangular distribution random variable. These values, named `duration_1`, `duration_2`, ..., `duration_9`, correspond to nine activities studied.
2) The completion duration values (`duration_i`) in step 1 are allocated to trajectories labeled `traj_1`, `traj_2`, ..., `traj_9`, which correspond to each activity in the PERT network. Trajectories trigger processes upon entity entry and are represented as global variables named `time_1`, `time_2` ..., `time_9`.
3) The activity paths in the PERT network are defined, with six paths used as references. Subsequently, six trajectory objects are designed, namely `tarj_148` referring to activity paths 1, 4, and 8, `traj_156`, ..., `traj_29`. Each path contains a timeout, determined by the global variable `time_i`. For example, the delay time of `traj_148` comprises global variables `time_1`, `time_4`, and `time_8`. The total time is stored in the attribute variable, `time_148`, with the same procedure applied to other trajectories. The attribute variable is used to attach the variable value to the entity.
4) The `traj_initial` trajectory is defined and acts as the initial point for the entity before proceeding to trajectories `traj_1`, `traj_2`, ..., `traj_9`. In `traj_initial`, an entity is first cloned to obtain six instances before simultaneously entering `traj_1`, `traj_2`, ..., `traj_9`. Therefore, the entities that enter these trajectories tend to initiate the process of assigning global values to the variable of `time_i`. For example, entry into `traj_1` initiates a global assignment of `time_1`.
5) The `traj_pert2` is similar to `traj_initial`, which duplicates the entity to six, corresponding with the number of paths in the network. Entities entering trajectory process timeouts to obtain the total. The resulting sum is stored in the attribute variable. For example, `traj_148` stores `time_148`, representing the total time it takes to complete paths 1, 4, and 8. The entity exiting `traj_pert2` has the highest attribute variable showing the longest time, stored as `execution_time`.
6) In the final stage of the simmer simulation environment (`PERT2`), two processes occur. The pro-

cesses are (1) an entity named dummy1, generated at time 0 to obtain `thetraj_initial`, and (2) dummy2, generated at time 0 in accordance with `traj_pert2`. However, this process is replicated 100 times.

The code is stated as follows:

```
# Process initialization
predict_WP <- function() {
   require(simmer)
   require(EnvStats)
   require(parallel)
   pert2 <- simmer("PERT2")
}

# Defining the distribution of activity time
duration_1 <- function() rtri(1, 1, 5, 3)
duration_2 <- function() rtri(1, 3, 9, 6)
duration_3 <- function() rtri(1, 10, 19, 13)
duration_4 <- function() rtri(1, 3, 12, 9)
duration_5 <- function() rtri(1, 1, 8, 3)
duration_6 <- function() rtri(1, 8, 16, 9)
duration_7 <- function() rtri(1, 4, 13, 7)
duration_8 <- function() rtri(1, 3, 9, 6)
duration_9 <- function() rtri(1, 1, 8, 3)
```

Each trajectory reads the duration of the corresponding activity by generating random variables based on the triangular distribution. In addition, the resulting values are stored globally.

```
# Process initialization
traj_1 <- trajectory() %>%
 set_global("time_1", duration_1)
traj_2 <- trajectory() %>%
 set_global("time_2", duration_2)
traj_3 <- trajectory() %>%
 set_global("time_3", duration_3)
traj_4 <- trajectory() %>%
 set_global("time_4",duration_4)
traj_5 <- trajectory() %>%
 set_global("time_5", duration_5)
traj_6 <- trajectory() %>%
 set_global("time_6", duration_6)
traj_7 <- trajectory() %>%
 set_global("time_7", duration_7)
traj_8 <- trajectory() %>%
 set_global ("time_8", duration_8)
traj_9 <- trajectory() %>%
 set_global("time_9", duration_9)
```

The subsequent section defines the trajectory in the PERT network. In this case, a total of six paths are obtained as shown in Fig. 2, namely: 1) paths 1-4-8, including activities 1, 4, and 8, 2) paths 1-5-6, 3) paths 1-5-7-9, 4) paths 2-6, 5) paths 2-7-9, and 6) paths 3-9. Each trajectory is named accordingly to represent the respective path. For example, `traj_39` is the trajectory for paths 3-9. R code for this section is stated as follows.

```
traj_148 <- trajectory() %>%
 timeout_from_global("time_1") %>%
 timeout_from_global("time_4") %>%
 timeout_from_global("time_8") %>%
 set_attribute("time_148",
    function() now(pert2))

traj_156 <- trajectory() %>%
 timeout_from_global("time_1") %>%
 timeout_from_global("time_5") %>%
 timeout_from_global("time_6") %>%
 set_attribute("time_156",
    function() now(pert2))

traj_1579 <- trajectory() %>%
 timeout_from_global("time_1") %>%
 timeout_from_global("time_5") %>%
 timeout_from_global("time_7") %>%
 timeout_from_global("time_9") %>%
 set_attribute("time_1579",
    function() now(pert2))

traj_26 <- trajectory() %>%
 timeout_from_global("time_2")\ %>%
 timeout_from_global("time_6")\ %>%
 set_attribute\ (\ "time_26",
    function() now(pert2))

traj_279 <- trajectory() %>%
 timeout_from_global("time_2") %>%
 timeout_from_global("time_7") %>%
 timeout_from_global("time_9") %>%
 set_attribute("time_279",
    function() now(pert2))

traj_39 <- trajectory() %>%
 timeout_from_global("time_3") %>%
 timeout_from_global("time_9") %>%
 set_attribute("time_39",
    function() now(pert2))
```

In the following section, the trajectory of an entity is established to initiate the generation of a triangular distribution random variable and store it globally. The trajectory ranges from `traj_1` to `traj_9`. In addition, `traj_initial` uses a simmer clone function to duplicate the entity into nine identical ones. Then, each entity enters the respective trajectory of `traj_1, traj_2, ..., traj_9`. The code for this process is stated as follows.

```
traj_initial <- trajectory() %>%
 clone(
     n = 9,
     traj_1,
     traj_2,
     traj_3,
     traj_4,
     traj_5,
     traj_6,
     traj_7,
     traj_8,
     traj_9
     )
```

This trajectory is designed to produce a global variable of `time_i` , representing the activity time across all PERT paths, ensuring uniformity. The simmer function `time_out_from_global` is employed for this purpose. The following code outlines the trajectory of the entity responsible for recording the completion duration of each particular PERT path. The set attribute function is `traj_pert2`.

```
traj_pert2 <- trajectory() %>%
 clone(
      n = 6,
      traj_148,
      traj_156,
      traj_1579,
      traj_26,
      traj_279,
      traj_39
      ) %>%
      synchronize(wait = TRUE) %>%
        set_attribute("execution_time",
           function() now(pert2))
```

The predicted completion time for the project is stored in the `execution_time` variable. Simulation must be run multiple times to obtain statistical measures, such as the standard deviation, standard error, confidence interval, and others. The available mclapply function in the parallel package is also used to obtain statistical measures. The `add_generator` function generates an entity named dummy1, which traverses the `traj_initial`. The second `add_generator` function is designed to generate an entity named `dummy2`, which traverses the `traj_pert2`.

```
# Simulation run 100 times
pert2 <- mclapply(1:100, function(i) {
 simmer("PERT2") %>%
 add_generator("dummy1",
   traj_initial, at(0), mon=2) %>%
 add_generator("dummy2",
   traj_pert2, at(0), mon=2) %>%
 run() %>% invisible %>%
 wrap()
})
```

The final part of this program is the functions to retrieve the simulation results, which are in the form of an R object with data.frame class or data sets.

```
has <- get_mon_attributes(pert2)
 result <- subset(has,
   key == "execution_time",
   select = c(key, time, replication))
 return(result)
}
```

The user-defined function is executed as follows:

```
dt <- prediction_WP()
  head(dt) # displays the first 6 lines
```

`Execution_time` is the longest duration among six paths in the PERT network, acting as a predictive variable for the duration of the project. This value is derived from the time variable. The replication variable refers to the repetition of simulation, comprising a total of 100 simulations, with only the first six replicates shown. The result is as follows.

	key	time	replication
16	execution_time	23.27893	1
32	execution_time	21.92817	2
48	execution_time	24.50765	3
64	run_time	21.06978	4
80	execution_time	25.78251	5
96	run_time	24.45695	6

The histogram representing the predicted project completion time is obtained as follows:

```
win.graph (width=4.6,height=5,
                              pointsize=10)
hist( dt$time,main="Prediction
\n ProjectCompletionTime",
xlab="CompletionTime")
```

The histogram in Fig. 3 shows a distribution close to normal. A normality test is carried out in accordance with the Shapiro-Wilk normality test and quantile-quantile (q-q) plots to validate this observation. The variance value of 0.3704 is quite large, showing that the time spread adhered to a normal distribution. It is reinforced by q-q plots.

```
shapiro.test(dt$time) # Normality test
        Shapiro-Wilk normality test
data: dt$time
W = 0.98594, p-value = 0.3704
```

The q-q plot in Fig. 4 confirms that the simulation data for project completion predictions are normally distributed [24, 25]. It can be understood because the sample size is quite large (100) and is obtained from the sum of the triangular distributions. The mean and standard deviation of these predictions are as follows.

```
(SDev <- sd(dt$time)) # Std Deviation
2.239672
(average <- mean(dt$time)) # average
20.69473
```

Based on simulation results, the average project can be completed in 20.7 weeks with a standard deviation of 2.24 weeks. Therefore, a 95% confidence interval estimator for the average project completion time is obtained. The average duration for the actual project completion, with a probability of 0.95, is between 20.2 to 21.1 weeks. The calculation is shown as follows.

```
SK95 <- average + c(-1, 1) * 2 * Sdev/10
20.24680 21.14267
```

Next, the critical path in the PERT network, relevant for determining the project completion time, is identified as the path with the longest duration. Therefore, the critical path of six pathways in the PERT network is obtained as follows:

```
dt2 <- prediksi_WP2()
maxTime2 <- tapply(dt2$time,
  factor(dt2$replication), max)
k <- as.numeric(names(maxTime2))
y <- data.frame(key="", time=0, replication=0)
for (j in k) {
```

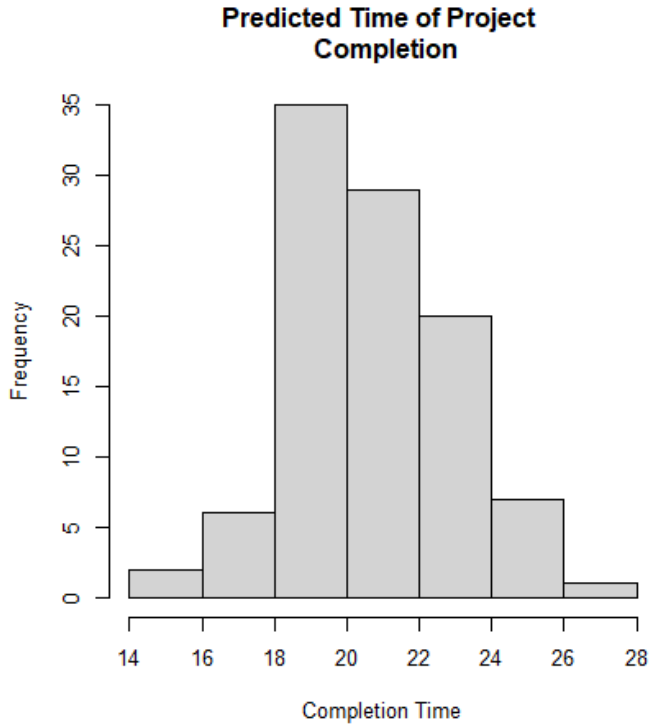

Fig. 3. Histogram of the predicted time of project completion.

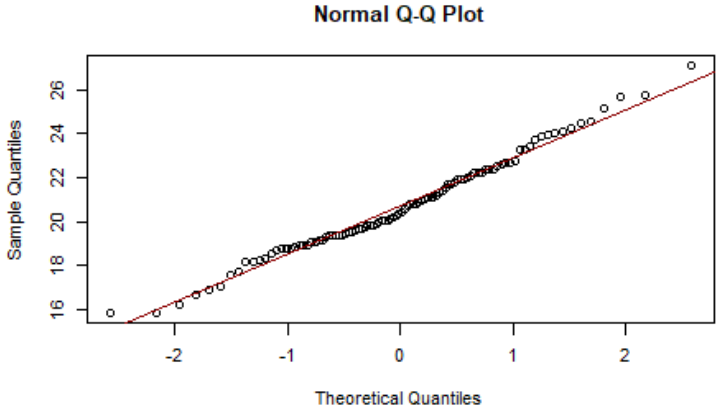

Fig. 4. Plot of quantiles.

```
 x = subset(dt2, replication == j,
     select = c(key, time, replication))
 y = rbind(y, subset(x,
     time == maxTime2[j],
       select = c(key, time, replication)))
}
ydat2 <- y[-1,]
head(ydat2) # displays 6 lines
```

The result is as follows.

key	time	replication
time_148	19.0	1
time_279	24.2	2
time_26	22.6	3
time_1579	21.2	4
time_39	21.8	5
time_148	19.3	6
...	...	...

The result shows that in the first replication, paths 1-4-8 are the longest, with 19.0 weeks. Then, in the second replication, paths 2-7-9 are the longest, and so on for other replications. Therefore, the longest

path is not similar for each replicated simulation. The following frequency table shows that path 1-5-7-9 is often the longest. Among the 100 replications of the simulations, path 1-5-7-9 is recorded 25 times as the longest path, while path 3-9 is recorded 19 times as the longest path.

```
table(factor(ydat2$key))
```

Path	Frequency	Freq.Relative
time_148	15	0.15
time_156	15	0.15
time_1579	25	0.25
time_26	9	0.09
time_279	17	0.17
time_39	19	0.19

In PERT, the main concern is the risk of not meeting the scheduled completion time promised to the project owner. The project completion time, as shown previously, is at least approximately normal distribution. The following is a project risk level at various promised completion times.

```
Time <- seq(21, 27, 0.5)
risk <- pnorm(Time, mean=average,
   sd=SDev, lower.tail = FALSE)
names(risk) <- Time
round(risk,3)
```

The result is as follows.

21	21.5	22	22.5	23	23.5
0.446	0.360	0.280	0.210	0.152	0.105
24	24.5	25	25.5	26	26.5
0.070	0.045	0.027	0.016	0.009	0.005
27					
0.002					

If the promised completion time is 25 weeks, the risk level is 2.7%. In other words, the probability that the project is completed beyond 25 weeks is 2.7%. However, if the maximum risk tolerance is 1%, the project manager should promise a maximum completion time of 26 weeks. The calculation is shown as follows.

```
(WS <- qnorm(0.01, mean=mean2,
       sd=SDev, lower.tail =FALSE))
25.90499
```

Simulation is rerun by developing DES using a simmer package. Each activity is represented as a timeout function in simulation without using resources to eliminate the queuing subsystem. Entities in this simulation traverse trajectory representing a path in PERT. In cases where the paths run parallel, entities are duplicated using the clone function on simmer. Consequently, identical entities traverse each parallel path, all starting at time 0. The total delay time in each path will be calculated and synchronized with the simmer function, which is synchronized with the argument function `wait=TRUE`. The function is paused until the entity exiting trajectory is the last one from the results duplicated, ensuring that the recorded time is the longest. It encapsulates the basic idea of developing a program for PERT simulation.

Simulation is repeated 100 times or as needed for further statistical analysis based on computational speed and specific requirements. In the research, 100 repetitions are presumed adequate. The simulation results are obtained as a data set retrieved from the output of the `get_mon_attributes` function. The resulting data set consists of three variables, namely key, time, and replication. The key variables contain the names of attributes or global variables stored during simulation. The time represents the delay time associated with each variable, while replication shows the sequence of simulation repetitions. The most important key value is `execution_time`, which stores the project completion time, as shown by the time value. This value is used to predict the completion time of a project for each replication. In addition, simulation saves the delay time for each path per iteration, enabling the determination of critical paths in each simulation replication. The analysis results show that the project completion time is statistically proven to be dispersed according to the normal distribution. Therefore, the project risk value can be calculated, representing the possibility of the project not meeting the predetermined completion timeframe.

IV. Conclusion

Simulation, including 100 replications, results in a predicted project completion time and a standard deviation of 20.7 and 2.2 weeks, respectively. A 95% confidence interval for the actual completion time falls in 20.2 to 21.1 weeks. In addition, path 1-5-7-9 is the longest path, identified as the critical path due to the longest completion time. When the promised completion time is 25 weeks, the associated risk level is 2.7%. Therefore, the probability that the project will be completed after 25 weeks is 2.7%. Assuming the tolerable risk has a maximum of 1%, the promised completion time of the project need not exceed 26 weeks.

The DES method has been proven highly effective in predicting the completion time of a project described by the PERT network. In addition, it offers increased flexibility because, after each completion of a project stage, the simulation model is rerun to obtain a more accurate prediction of the completion time. The inputs in this simulation include 1) the activities in the project of concern, 2) the sequence of these activities, and 3) the estimated fastest, most prolonged, and most frequent completion time or mode for each activity. These

inputs, in addition to the completion and terminated statuses of each activity, form the PERT network.

The research has the potential for further development to increase the usefulness for end-users not proficient in R. One avenue for the extension is the design of web-based applications. These applications enable users to input and update parameters based on actual completion times of project activities. The applications generate simulation outputs by incorporating real-time data updates, including project completion predictions and associated risks, which closely reflect actual values. Therefore, this application is expected to help oversee and control ongoing projects. The implementation of this idea with the methods currently available tends to have posed challenges.

ACKNOWLEDGEMENT

The research receives funding support from the Research and Technology Transfer Office of Bina Nusantara University under the auspices of the university's Serving the Nation research grant. The grant explicitly supports the project titled "*Pengembangan Paket R untuk Mendukung Pembelajaran Simulasi Kejadian Diskret*," as outlined in contract number 064/VR.RTT/IV/2022, signed on April 8, 2022.

AUTHOR CONTRIBUTION

Writing—original draft, I G. A. A. Y.; Methodology, I G. A. A. Y., and R. N.; Formal analysis, I G. A. A. Y., and R. N.; Analysis result review, I G. A. A. Y., R. N., S. A., A. A. A., P. W.P., and N. S. All authors have read and agreed to the published version of the manuscript.

REFERENCES

[1] A. M. Law, W. D. Kelton, and W. D. Kelton, *Simulation modeling and analysis*. McGraw-Hill, 2007, vol. 3.

[2] J. Banks, J. S. Carson, B. L. Nelson, and D. M. Nicol, *Discrete-event system simulation: Pearson new international edition*. Pearson Education, 2013.

[3] V. Vinod and R. Sridharan, "Simulation modeling and analysis of due-date assignment methods and scheduling decision rules in a dynamic job shop production system," *International Journal of Production Economics*, vol. 129, no. 1, pp. 127–146, 2011.

[4] A. A. B. Pritsker and J. J. O'Reilly, *Simulation with visual SLAM and AweSim*. John Wiley & Sons, 1999.

[5] A. Ebert, P. Wu, K. Mengersen, and F. Ruggeri, "Computationally efficient simulation of queues: The R package queuecomputer," 2017. [Online]. Available: https://arxiv.org/abs/1703.02151

[6] I. Ucar, B. Smeets, and A. Azcorra, "Simmer: Discrete-event simulation for R," *Journal of Statistical Software*, vol. 90, no. 2, pp. 1–30, 2019.

[7] J. Zuo, W. Q. Meeker, and H. Wu, "A simulation study on the confidence interval procedures of some mean cumulative function estimators," *Journal of Statistical Computation and Simulation*, vol. 83, no. 10, pp. 1868–1889, 2013.

[8] H. Wang and Z. J. Zhai, "Advances in building simulation and computational techniques: A review between 1987 and 2014," *Energy and Buildings*, vol. 128, pp. 319–335, 2016.

[9] B. Smeets and I. Ucar, "Introduction to simmer," 2023. [Online]. Available: https://cloud.r-project.org/web/packages/simmer/vignettes/simmer-01-introduction.html

[10] I. G. A. A. Yudistira, "Pengembangan simulasi kejadian diskret berbasis paket simmer pada R," *Engineering, MAthematics and Computer Science (EMACS) Journal*, vol. 3, no. 2, pp. 79–85, 2021.

[11] I. Ucar, J. A. Hernández, P. Serrano, and A. Azcorra, "Design and analysis of 5G scenarios with simmer: An R package for fast des prototyping," *IEEE Communications Magazine*, vol. 56, no. 11, pp. 145–151, 2018.

[12] M. Lu and S. M. AbouRizk, "Simplified CPM/PERT simulation model," *Journal of Construction Engineering and Management*, vol. 126, no. 3, pp. 219–226, 2000.

[13] W. N. Shofa, I. Soejanto, and T. Ristyowati, "Penjadwalan proyek dengan penerapan simulasi Monte Carlo pada metode Program Evaluation Review and Technique (PERT)," *Opsi*, vol. 10, no. 2, pp. 150–157, 2017.

[14] N. Ljiljanić, Z. Rajić, and T. Paunović, "Use of PERT (Program Evaluation and Review Technique) and PDM (Precedence Diagramming Method) in organizing modern vegetable seedling production," *Ekonomika Poljoprivrede*, vol. 69, no. 1, pp. 119–131, 2022.

[15] M. C. Sachs and E. E. Gabriel, "Event history regression with pseudo-observations: Computational approaches and an implementation in R," *Journal of Statistical Software*, vol. 102, pp. 1–34, 2022.

[16] V. Knight and G. Palmer, *Applied mathematics with open-source software: Operational research problems with Python and R*. CRC Press, 2022.

[17] S. Venturini and R. Piccarreta, "A Bayesian approach for model-based clustering of several binary dissimilarity matrices: The dmbc package in

R," *Journal of Statistical Software*, vol. 100, pp. 1–35, 2021.

[18] S. J. Eglen, "A quick guide to teaching R programming to computational biology students," *PLoS Computational Biology*, vol. 5, no. 8, pp. 1–4, 2009.

[19] A. P. Goldberg and J. R. Karr, "DE-Sim: An object-oriented, discrete-event simulation tool for data-intensive modeling of complex systems in Python," *Journal of Open Source Software*, vol. 5, no. 55, pp. 1–7, 2020.

[20] S. P. Millard, A. Kowarik, and M. A. Kowarik, *Package 'EnvStats'*, 2018.

[21] I. G. A. A. Yudistira, R. Nariswari, and S. Arifin, "output visualization from result of discrete event system simulation with 'simmer' R package," *BAREKENG: Jurnal Ilmu Matematika dan Terapan*, vol. 17, no. 1, pp. 0581–0592, 2023.

[22] X. Dong, L. Castro, and N. Shaikh, "Fastnet: An R package for fast simulation and analysis of large-scale social networks," *Journal of Statistical Software*, vol. 96, pp. 1–23, 2020.

[23] P. C. Jiménez and Y. R. Montoya, "Queueing: A package for analysis of queueing networks and models in R," *R Journal*, vol. 9, no. 2, 2017.

[24] P. O. Farayola, S. K. Chaganti, A. O. Obaidi, A. Sheikh, S. Ravi, and D. Chen, "Quantile–quantile fitting approach to detect site to site variations in massive multi-site testing," in *2020 IEEE 38th VLSI Test Symposium (VTS)*. San Diego, USA: IEEE, April 5–8, 2020, pp. 1–6.

[25] T. J. Cleophas and A. H. Zwinderman, "Quantile-quantile plots, a good start for looking at your medical data (50 cholesterol measurements and 58 patients)," *Machine Learning in Medicine–A Complete Overview*, pp. 319–327, 2020.

APPENDIX

https://github.com/Anom1392/PERT - Complete R code generated by the research.