

GECCO 2021 Industrial Challenge: *Optimization of a simulation model for a capacity and resource planning task for hospitals under spe- cial consideration of the COVID-19 pandemic*

Frederik Rehbach, Margarita Rebolledo, Sowmya Chandrasekaran,
Thomas Bartz-Beielstein ¹

February, 2021

The goal of the GECCO 2021 Industrial Challenge is to develop an optimizer for this year's industrial application in a programming language of your choice. The optimizer has to efficiently use every objective function evaluation as each evaluation involves a computationally intensive discrete event simulation.

The challenge is split into two tracks. In the first track, the amount of allowed objective function evaluations is only limited by the computing power you are willing to invest on your own machine. In the second track, only 200 evaluations are allowed per optimization run. In both cases, the algorithm with the best-found objective function value wins.

This document provides a set of rules and regulations for the GECCO IC, a detailed problem description, as well as contact and submission information.

¹ Cologne University of Applied Sciences, 51643 Gummersbach, Germany
frederik.rehbach@th-koeln.de,
thomas.bartz-beielstein@th-koeln.de,
margarita.rebolledo@th-koeln.de,
sowmya.chandrasekaran@th-koeln.de

Technology
Arts Sciences
TH Köln

1 Introduction

THE GOAL of the GECCO 2021 Industrial Challenge is to develop an optimization algorithm for the resource and capacity simulator BabSimHospital. The simulator was developed in cooperation with intensive care unit (ICU) experts, crisis teams and health administrations. It is introduced in ² and ³.

The BabSimHospital is a 29-dimensional discrete event simulation (DES) problem. Each objective function evaluation simulates the paths that COVID-19 infected patients follow during their hospital stays. An initial estimation of the simulator parameters is specified in cooperation with medical professionals.

The aim of the challenge is to allow for an open optimization process, in the sense that, the competitors are allowed to use any programming language and optimization technique of their choice.

HIGHLIGHTS of the GECCO IC include:

- Interesting Problem Domain: The impact of COVID-19 on the health system is ongoing and tools to help in capacity planning are more important than ever.
- Real-world Problems: Test your algorithms and methods, directly on real and current data.
- Easy Access: Easily Participate through our online platform, no installations required.

² Thomas Bartz-Beielstein, Frederik Rehbach, Olaf Mersmann, and Eva Bartz. Hospital capacity planning using discrete event simulation under special consideration of the covid-19 pandemic, 2020b

³ Thomas Bartz-Beielstein, Eva Bartz, Frederik Rehbach, and Olaf Mersmann. Optimization of high-dimensional simulation models using synthetic data, 2020a

- Fair Submission Assessment: Winners are determined automatically through our online portal, fully objectively, only based on the final result quality.
- Publication Options: Participants can submit a 2-page extended abstract describing the algorithm they are applying in the challenge. Accepted abstracts are published together with GECCO poster-papers in the GECCO-companion.
- Price Money: The top three participants of the limited track will be honored with price money.

The remainder of this document specifies the information needed to take part in this competition. Section 2 gives more details about the simulator. Section 3 summarizes how to participate in the challenge and which rules have to be followed.

2 Problem Description

2.1 Hospital Capacity and Resource Planning Using Discrete Event Simulation

Resource and capacity planning with respect to the COVID-19 pandemic is a challenging task for hospitals. BabSimHospital is a resource planning tool developed in cooperation with ICU experts, crisis teams and health administrations. The core of BabSimHospital is a DES that simulates the typical paths COVID-19 patients follow in their hospital stays, from infection to recovery/death. The simulator models the number of ICU beds as function of the number of infected individuals.

The simulator requires data that, on the one hand, describes the spread of the pandemic over time, and on the other hand, includes the resource usage, e.g., the number of available ICU beds or the number of patients on ventilators. The necessary data is automatically obtained from the Robert Koch Institute ⁴ (RKI) and the Deutsche interdisziplinäre Institute ⁵ (DIVI) daily reports. The associated risk like age and gender are also obtained in this stage.

⁴ <https://www.rki.de>

⁵ <https://www.divi.de>

The path or trajectory an infected patient can follow is illustrated in Figure 1. Each node in the figure represents one hospital station. The edges between different stations represent the associated probability and duration of the station change. There are currently 29 parameters.

BabSimHospital provides a default parameter set which can be used for simulations. However, time-dependent changes require a frequent refitting of the model parameters, e.g., the increase in rate of successful treatments in Germany. These parameters are also dependent on the geographical location of the patient. Proper tuning of the model parameters is essential to obtain an accurate prediction of the resource usage. **Finding the optimal values for these parameters is your task in this challenge.** A short overview of all 29 parameters and their recommended ranges is given in Table 1.

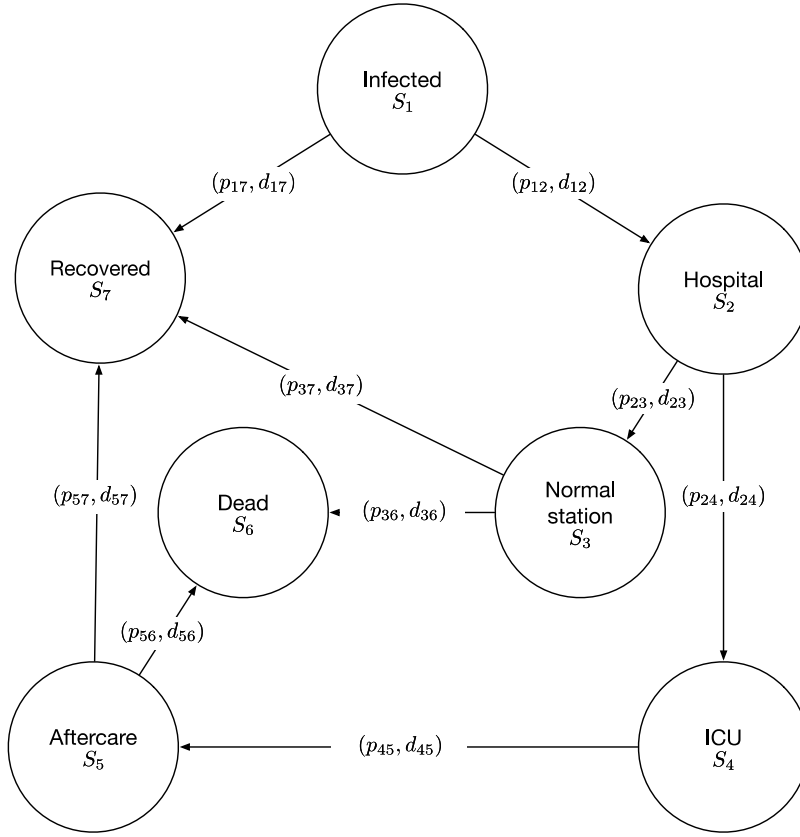


Figure 1: Simplified model of a patient flow in hospital. Nodes represent states(S_i). Edges represent the associated probabilities (p_{ij}) and duration (d_{ij}). Probability and duration form the model parameters.

3 How to Participate

There are two tracks in this competition. Competitors can participate in both tracks, but can also just choose one of them. For the purpose of this challenge a Docker image was created containing all the necessary data. Docker is an open-source tool that provides linux container-based operating system-level virtualization⁶. Docker can be easily installed on any major operating system. It allows for packaging code or experiments into a shareable container image. An image typically contains all required software and prerequisites to run a certain task. Such images can be shared and be ready for immediate execution on other machines. For more information and installation you can visit the Docker website <https://www.docker.com/get-started>

⁶ Carl Boettiger. An introduction to docker for reproducible research. *ACM SIGOPS Operating Systems Review*, 49(1):71–79, 2015

Track 1: Unlimited Evaluations

In track one the amount of objective function evaluations is only limited by how much computation time you are willing to give to the optimizer on your machine. In order to participate you will have to download the corresponding Docker image (may need sudo rights to download). Currently, there are only Python and R examples of how to run the BabSimHospital simulator locally. However, any programming language that can access the command

line on your machine should be capable of using the easy interface.

On your terminal console an evaluation in the BabSim.Hospital should look something like this (take care, the formatting of the symbols - and ' can cause this command not to work on your terminal):

```
docker run --rm mrebolle/r-geccoc:Track1 -c 'Rscript objfun.R "6,7,3,3,3,5,3,3,25,
17,2,1,0.25,0.05,0.07,0.005,0.07,1e-04,0.08,0.25,0.08,0.5,1e-06,2,1e-06,1e-06,1,2,0.5"'
```

Once you have setup the evaluation interface on your machine, you can try to use any optimizer of your choice to find the lowest objective function value on BabSim.Hospital. Example scripts for that are available in the folder "DockerExampleScripts".

| Parameter | Range |
|---|--------------|
| AmntDaysInfectedToHospital | 6-14 |
| AmntDaysNormalToHealthy | 7-13 |
| AmntDaysNormalToIntensive | 3-7 |
| AmntDaysNormalToVentilation | 3-9 |
| AmntDaysNormalToDeath | 3-7 |
| AmntDaysIntensiveToAftercare | 5-9 |
| AmntDaysIntensiveToVentilation | 3-5 |
| AmntDaysIntensiveToDeath | 3-7 |
| AmntDaysVentilationToIntensiveAfter | 25-35 |
| AmntDaysVentilationToDeath | 17-25 |
| AmntDaysIntensiveAfterToAftercare | 2-5 |
| AmntDaysIntensiveAfterToDeath | 1-7 |
| GammaShapeParameter | 0.25-2 |
| FactorPatientsInfectedToHospital | 0.05-0.15 |
| FactorPatientsHospitalToIntensive | 0.07-0.11 |
| FactorPatientsHospitalToVentilation | 0.005-0.02 |
| FactorPatientsNormalToIntensive | 0.07-0.13 |
| FactorPatientsNormalToVentilation | 1e-04-0.002 |
| FactorPatientsNormalToDeath | 0.08-0.12 |
| FactorPatientsIntensiveToVentilation | 0.25-0.35 |
| FactorPatientsIntensiveToDeath | 0.08-0.12 |
| FactorPatientsVentilationToIntensiveAfter | 0.5-0.9 |
| FactorPatientsIntensiveAfterToDeath | 1e-06-0.01 |
| AmntDaysAftercareToHealthy | 2-4 |
| RiskFactorA | 1e-06-1.1 |
| RiskFactorB | 1e-06-0.0625 |
| RiskMale | 1-2 |
| AmntDaysIntensiveAfterToHealthy | 2-5 |
| FactorPatientsIntensiveAfterToHealthy | 0.5-0.75 |

Table 1: Overview of the 29 simulator parameter with their recommended ranges

Listing 1: 'example.py': sample Python code that creates a random candidate solution and evaluates it on the BabSim.Hospital problem.

```
import subprocess
import random

## evalFun accepts a vector of integers ,
## length 29 (dimensionality of the BabSim.Hospital problem)
def evalFun(candidateSolution):
    evalCommand = "docker run --rm"
    mrebolle/r-geccoc:Track1 -c 'Rscript objfun.R "
    parsedCandidate = ",".join([str(x) for x in
                                candidateSolution])
    return(subprocess.check_output(evalCommand + "\"" +
                                parsedCandidate + "\"" + " ", shell=True))

## Create some candidate
lower = [6,7,3,3,3,5,3,3,25,17,2,1,0.25,0.05,0.07,0.005,0.07,
1e-04,0.08,0.25,0.08,0.5,1e-06,2,1e-06,1e-06,1,2,0.5]
upper = [14,13,7,9,7,9,5,7,35,25,5,7,2,0.15,0.11,0.02,0.13,0.002,
0.12,0.35,0.12,0.9,0.01,4,1.1,0.0625,2,5,0.75]

candidateSolution=[]
for x in range (29):
    candidateSolution.append(random.uniform(lower[x],upper[x]))

## Evaluate the candidate , for example on the BabSimHospital problem:
print(evalFun(candidateSolution))
```

Participation is then very easy: You can send your best candidate setup (the 29 parameter vector) along with your measured objective function value to "gecco@gm.fh-koeln.de", mail-subject "GECCO-IC-2021". Each candidate that we receive will be checked at the end of the competition, and the winners will be determined according to the objective function value that we measure.

Finalists selected by the organizers will be invited to present their submission at the competition session, held during the GECCO conference. The winner of the competition will be announced at the SIGEVO meeting ceremony, on July 2021.

Track 2: Severely Limited Evaluations

For the second track, you do not have to install any additional software on your machine. Submissions will be handled through an automated online evaluation tool. You can access the tool via:

<http://owos.gm.fh-koeln.de:3838/GECCO-IC-2021/>

The entrance password to the tool is IC2021. After entering the page you will have to create an account on the registration page. Please remember your password as we have NO mechanisms for resetting a forgotten password! Once you created an account, you can start uploading your submissions as often as you like. The submissions and scores will be saved on our servers.

A submission to the online tool will consist of multiple files (You have to mark all your files simultaneously after you pressed the "browse" button). Each upload has to contain one file called "main.sh". The evaluation tool will search for this exact file name

and try to run it for your evaluation.

The dockerized environment that your code will be run in contains installations for python 2.7.18 & 3.8.5 as well as R 4.0.3. If you want to use other programming languages or install other prerequisites for your optimizers you can do so. You are in a simplified Ubuntu environment. Therefore, commands like "apt-get install" etc. can be used to add additional software. In the container you are the root user, so using the sudo command is not required. You have to specify each of your prerequisite installation steps as well as lastly the call to your optimizers source code in the "main.sh".

In the following we give a short example of how to setup a very simple optimizer that tries a few random candidates in python. These codes and other examples are included in the ExampleScripts folder of the resource package. In order to try the base example online, upload both the "main.sh" and "optimRandomSearch.py" to the webpage. After some time (press refresh from time to time), you will see the evaluation result.

Since python is already installed in the docker container, our "main.sh" only contains a single line:

Listing 2: Code inside example 'main.sh'. The script is automatically started on the evaluation server and runs our optimizer.

```
python3 optimRandomSearch.py
```

Listing 3: 'optimRandomSearch.py': sample Python code that creates random candidate solutions and evaluates them on the BabSim.Hospital problem.

```
import subprocess
import random

## evalFun accepts a vector of integers ,
## length 29 (dimensionality of the BabSim.Hospital problem)
def evalFun(candidateSolution):
    evalCommand = "./evaluate.sh "
    parsedCandidate = ",".join([str(x) for x in
                                candidateSolution])
    return(subprocess.check_output(evalCommand + "'" +
                                   parsedCandidate + "'", shell=True))

for i in range(5):
    ## Generate a bunch of random candidate solutions
    candidate=[]
    for x in range(29):
        candidate.append(random.uniform(lower[x],upper[x]))
    ## Evaluate the candidates
    evalFun(candidate)
```

Listing 3 presents a python example code that can be uploaded to the online submission tool. Similar to the requirements in track 1, the BabSimHospital problem is called via the command line. The objective function "evalFun" accepts a vector with 29 floating point values and passes them via the command line to 'evaluate.sh'. This same approach has to be used in any other programming language too. 'evaluate.sh' accepts a comma separated string of numeric

values e.g. "5,1,14,0.08,...", evaluates the candidate and returns the objective function value. Your script / optimizer does not have to return anything. The best evaluated candidate is automatically stored on the server and returned as your result.

Each uploaded algorithm should only use 200 evaluations and can not use information gained from previous runs. In order to ensure this, the online platform applies each uploaded algorithm to a newly created randomized instance of the BabSim.Hospital problem.

For the final evaluation of the winner a different undisclosed geographical location will be used for every participant to keep the comparison fair. The LAST UPLOADED algorithm of each participant will be applied to that new location. The algorithm with the single best-found candidate wins.

Finalists selected by the organizers will be invited to present their submission at the competition session, held during the GECCO conference. The winner of the competition will be announced at the SIGEVO meeting ceremony, on July 2021. While highly appreciated, it is not required for the participants of the challenge to also participate in the GECCO conference.

3.1 Organizing Committee

If you have questions, feel free to reach out to us at "gecco@gm.fh-koeln.de", or personally:

- Frederik Rehbach, TH Köln - frederik.rehbach@th-koeln.de
- Margarita Rebolledo, TH Köln - margarita.rebolledo@th-koeln.de
- Sowmya Chandrasekaran, TH Köln - sowmya.chandrasekaran@th-koeln.de
- Thomas Bartz-Beielstein, TH Köln - thomas.bartz-beielstein@th-koeln.de

List of References

Thomas Bartz-Beielstein, Eva Bartz, Frederik Rehbach, and Olaf Mersmann. Optimization of high-dimensional simulation models using synthetic data, 2020a.

Thomas Bartz-Beielstein, Frederik Rehbach, Olaf Mersmann, and Eva Bartz. Hospital capacity planning using discrete event simulation under special consideration of the covid-19 pandemic, 2020b.

Carl Boettiger. An introduction to docker for reproducible research. *ACM SIGOPS Operating Systems Review*, 49(1):71–79, 2015.