Business Process Optimization Competition ‘24

Overview

The objective of the *BPO Competition 2024* is to build a patient planning mechanism for an artificial hospital. To do this, the competitors implement a ‘plan’ function that assigns patients to the moment in time at which they will be treated.

During opening hours, two types of patients can arrive at the hospital:

1. Planned patients who have been given an appointment by your implemented scheduling tool.
2. Emergency patients show up randomly throughout the day.

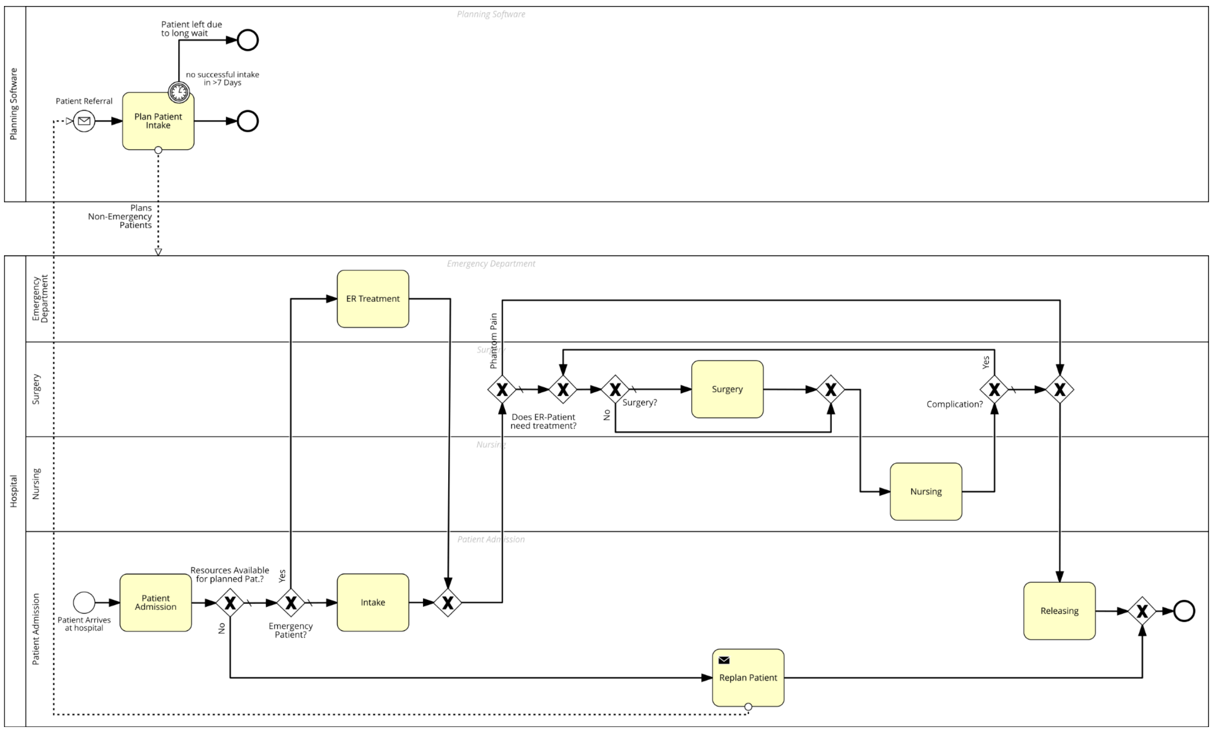

Figure 1. Hospital’s processes

Figure 1 shows the hospital’s processes. The competitors are supposed to implement the ‘*Plan Patient Intake*’ activity. *Planned* and *emergency (ER) patients* arrive and are admitted.

Emergency patients

Upon arrival, *ER patients* will immediately proceed to the *Emergency Department* and subsequently receive *Emergency Treatment*. For some of the *ER patients,* further treatment in the *Surgery* or *Nursing* department will be necessary. These patients will be diagnosed with a case, while others can be released after having received the *Emergency Treatment*. The *ER patients* for which further treatment is necessary will receive treatment, e.g., surgery, with a higher priority than the planned patients. Once the treatment has been completed, they will be released.

Planned patients

Planned patients are referred to the hospital with a known diagnosis. They must be assigned a time slot on which they will subsequently arrive to receive their treatment. When they arrive at the hospital, and it is apparent that timely treatment is infeasible, they will be sent home right away. Treatment is infeasible either if more than two patients have finished the intake, but have not yet been processed in the *Surgery* or *Nursing* departments, or when all Intake resources are occupied upon a patient's arrival. If treatment is feasible, the planned patients will proceed to *Intake*. Depending on the treatment case, some patients need Surgery, while others need only Nursing. After having finished their treatment, they will be released. For both patient groups complications during their treatments can arise, requiring additional surgery or nursing.

Diagnoses

Diagnoses are either of type A or B. The diagnoses are further subdivided into subgroups 1 to 4. Thus, a diagnosis can be identified as, e.g., A1, B3, …

Resource Availabilities

Intake: Four intake personnel are available on Monday-Friday from 8 a.m. until 5 p.m. Any intake resource will take norm(1, ⅛) hours to conduct the intake of any patient.

Surgery: Five operating rooms are available during working hours. This drops to one operating room outside of the working hours. The surgery times depend on the cases and are listed in the table below.

Nursing: Two types of nursing wards are available, one for type A patients and one for type B patients. The ward for type A patients has a capacity of 30 beds, and the ward for type B patients of 40 beds. The nursing times depend on the cases and are listed in the table below.

Emergency Department: Nine personnel are available for conducting the *ER treatment*. Any ER resource will take norm(2, ½) hours to conduct the ER treatment of any patient.

Patients arrival rates

Patient type	Rate	Diagnosis (Probability)	Operation time	Nursing time	Complications (Probability in %)
A	unif(0,1)	A1 (1/2)	-	norm(4, 1/2)	1%
		A2 (1/4)	norm(1, 1/4)	norm(8, 2)	1%
		A3 (1/8)	norm(2, 1/2)	norm(16, 2)	2%
		A4 (1/8)	norm(4, 1/2)	norm(16, 2)	2%
B	unif(0,1)	B1 (1/2)	-	norm(8, 2)	0.1%
		B2 (1/4)	-	norm(16, 2)	1%
		B3 (1/8)	norm(4, 1/2)	norm(16, 4)	2%
		B4 (1/8)	norm(4, 1)	norm(16, 4)	2%
EM	exp(1)	*			

Competition Objective

The decision that must be made is which patient to admit at which time. Patients must be planned for a timeslot 24 hours before their hospitalization. When either no more than two patients have finished the intake but have not further been processed in the *Surgery* or *Nursing department*, or when all Intake resources are occupied upon a patient's arrival, they are sent away and can be planned for another timeslot.

Planned patients who have not successfully been taken in within 7 days after their referral will go to another hospital.

Performance Measurement and Scoring

The objective is to optimize the planning according to the following criteria:

er_treatment_score
ER patients who require further treatment should be processed within 4 hours of time.
Hence, every ER patient who does violate that requirement incurs a penalty. The penalty is $(t - 4)^2$, where t is the time in hours from the end of the ER Treatment until the start of the next activity, i.e., *Nursing* or *Surgery*.

sent_home_score
Sending planned patients home contributes to a patient's dissatisfaction. Every patient that has been sent home will incur a cost.

processed_score
The hospital's main objective is to process as many patients as possible. Hence, the processed_score is a penalty for every case that could not be processed, i.e., because patients left because they could not be taken in within 7 days after their referral. For the final

ranking, all three scores will be min-max normalized over the range of scores of the different participants. The average of the three min-max normalized scores will determine the final ranking.

Technical Details

You can run the simulator using:

```
problem = HealthcareProblem()
simulator = Simulator(your_planner, problem)
result = simulator.run(365*24)
```

In this code your_planner is an instance of a class that you implement yourself. This class must subclass planners. Plan and implement the following function.

```
def plan(self, plannable_elements, simulation_time):
```

A function that must plan patient activities for a particular time. In particular, it must plan the moment at which patients arrive for their intake. The function takes a parameter plannable_elements, which is a dictionary with case identifiers (i.e., patient numbers) as keys and for each key as value the list of events that must be planned for the patient. For this competition, the value will only ever be time_for_intake. The function also takes the current simulation_time as a parameter.

The function must return a list of triples (<case_id>, <plannable_element>, <time>), which represent the times at which each patient's intake must be planned.

For example, a function call that could be expected is

```
plan({3: ‘time_for_intake’}, 2)
```

which represents that there is a single patient with identifier 3 for whom the time_for_intake must be planned, while the current time is 2. A valid return for this request would be

```
[(3, ‘time_for_intake’, 33.5)]
```

which represents that the time_for_intake of the patient with identifier 3 is planned at time 33.5.

Note that the time is measured in hours from Monday 1 January 2018 at 00:00.

In addition to this, you can also implement the following function.

```
def report(self, case_id, element, timestamp, resource, lifecycle_state):
```

A function that you can use to collect information from the simulator. It is called each time one of the following happens: a new case arrives, a task becomes ready to perform, performing a task started, performing a task completed, a case completes. The simulator does not expect any return value. This function is just there for you to collect data on what is happening in the process, so you can also have it do nothing.

There are two ready-made reporters that you can use for your convenience in the reporter.py library. One simply prints each event to the standard output, the other exports them as an event log that can be analyzed using your favorite process mining tool.

An example planner is implemented in the __example__.py file for your reference.

Important Notes

It is explicitly forbidden to use any variables that you did not create yourself, based on information that you obtained via the plan or report functions. Doing so will lead to a desk rejection of the submission.

Your code must simulate one year (365*24 hours) of cases within 2 hours on a typical desktop computer. Code that takes longer to run will be rejected.

By default, the simulator always simulates a year (365*24 hours) of cases. However, this may take quite long, which may be prohibitive when you are coding or debugging. For that reason, you can also pass the number of hours you want to simulate to the run method.

There is a planner helper available with some convenient functions that you can use. You can get your plannerhelper as self.planner_helper. The documentation is provided in plannerhelper.py.

Submission

Submit your solution by 15 August 2024 by e-mail to: bpo.competition@bpm.cit.tum.de
Send in your submission as a single .zip file that includes:

- Your code in Python. This can contain multiple files. Make sure there is one file that ends with:
  ```
  problem = HealthcareProblem()
  simulator = Simulator(your_planner, problem)
  result = simulator.run(365*24)
  ```
  This is the file from which we will call your code to evaluate your solution.
- A .pdf file with a brief description (max. 1 A4) of the main technique or techniques that you used for planning.
- A .txt file with the names, affiliations, and e-mail addresses of the people who make the submission.

Evaluation process

Your solution will be checked between 16 August and 2 September 2024. On 2 September 2024 the winner of the competition will be announced at the BPO workshop (https://sites.google.com/view/bpo2024/).