Simulation with Python

Xudong Wang, Xueping Li

[1]Department of Industrial & Systems Engineering, The University of Tennessee, Knoxville.

THE UNIVERSITY OF
TENNESSEE
KNOXVILLE

Why Simulation

- Modeling Flexibility: AnyLogic provides a graphical modeling environment that allows you to easily create complex models by dragging and dropping components, making it more accessible to users who may not be proficient in programming languages. This can speed up the development of simulation models.
- Multi-Paradigm Modeling: AnyLogic supports multiple modeling paradigms, which can be combined in a single simulation model. This allows you to represent a wider range of systems and processes, especially those that involve interactions between agents, continuous processes, and feedback loops.
- Pre-built Libraries: AnyLogic offers a library of pre-built simulation objects and templates for common modeling elements, such as queues, conveyors, and resource allocation. This can save time and effort in model development.
- Visualization
- Integration
- ...

Challenges in Simulation optimization

- Variable formulation - Ranking and Selection
  - Everything needs to be transferred into variables (discrete or continuous) for “OptQuest”
- Computational intensity
  - Too many variables will make it be time-consuming
  - Multiple simulations are needed in stochastic scenarios
- Limited exploration of solution space
  - The solution quality depends on the sampling method
- Curse of dimensionality
  - The complexity of exploring the solution space grows exponentially if the number of decision variables increases.

Purpose

Utilize the power of the simulation engine with existing models and existing machine learning libraries: // Why not all-in Python?

- Allow optimization of simulation models
- Avoid complexities of model translation between simulation and ML
- Improve the flexibility, effectiveness, and visualization

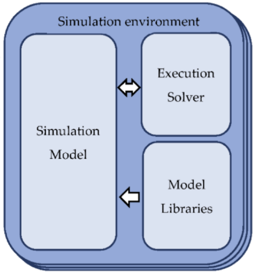


Figure: Optimization model & machine learning Simulation integration. [1]

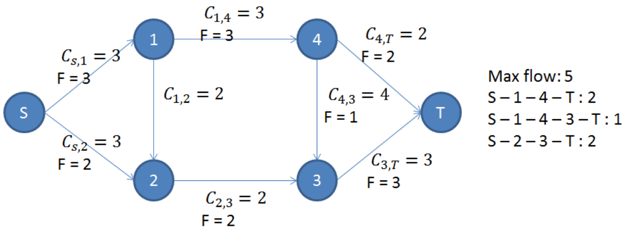
Pypeline

A custom library for AnyLogic that allows you to call Python within a running model. It connects to a local installation of Python, allowing you to make use of any Python library. It can be used for cases such as:

- Utilizing code that was originally written in Python without having to port to Java
- Writing complex algorithms in Python that you can call in Java, optionally passing objects/data between the languages
- Being able to use any Python-exclusive library

Maximum Flow

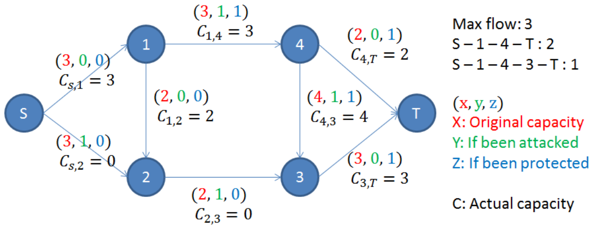
- The max flow problem is an optimization problem that involves finding the maximum flow that can be sent through a directed graph from a source vertex to a terminal vertex while taking into account the capacity limitations of each arc. [2]
- The concept of flow represents the movement of a resource, such as water through a network of pipes or data through a computer network. This concept finds wide application in various domains, including power grids and transportation networks, where the efficient allocation and distribution of resources are crucial.



Interdiction

Maximum Flow Network Interdiction

- Maximum Flow Network interdiction extends the basic max flow problem with the defender and attacker, which is meaningful in infrastructure protection, counter-terrorism, disaster response, emergency management, and supply chain optimization. [3]
- The defender will choose some arcs to protect them from being destroyed by the attacker. The objective is to maximize the maximum flow after interdiction.
- However, the attacker wants to interdict some arcs without the protection, in order to minimize the maximum flow in the graph after interdiction.



Assumptions

- The network has one resource node and one sink node.
- Both players will choose a limited number of arcs to attack/fortify.
- Both players will not know their enemy's strategies while making decisions.
- The game will start from the attacker's side and take turns between two players.
- The objective of the attacker is to maximize max-flow after interdiction while the defender aims to minimize max-flow after interdiction.

Notations

- Parameters:
  - C_{ij}: Flow capacity of the arc from vertex i to vertex j.
  - D: The maximum number of defended arcs.
  - A: The maximum number of attacked arcs.
  - α_{ij}: If arc i to j is attacked, $\alpha_{ij} = 1$; Otherwise $\alpha_{ij} = 0$.
  - β_{ij}: If arc i to j is fortified, $\beta_{ij} = 1$; Otherwise $\beta_{ij} = 0$.
- Variables:
  - f_{ij}: Actual flow from arc i to j after interdiction.
  - F: Maximum flow from S to T.
  - X_{ij}: If arc i to j is defended $X_{ij} = 1$; Otherwise $X_{ij} = 0$.

Max-flow after Interdiciton

$$
\begin{aligned}
\text{maximize} \quad & F \\
\text{subject to} \quad & \sum_i f_{Si} = F \\
& \sum_i f_{iT} = F \\
& \sum_i f_{ij} = \sum_i f_{ji}, \forall j, j \neq S, T \\
& 0 \leq f_{ij} \leq C_{ij} * Z_{ij} \\
& \sum_i \sum_j \alpha_{ij} \leq A \\
& \sum_i \sum_j \beta_{ij} \leq D \\
& \beta_{ij} - \alpha_{ij} \leq Z_{ij} \\
& \beta_{ij} - \alpha_{ij} + 1 \geq Z_{ij}
\end{aligned}
\tag{2.1}
$$

Reference I

[1] Changyong Chu, Chengfang Yin, Shaohui Su, and Chang Chen. "Synchronous Integration Method of System and Simulation Models for Mechatronic Systems Based on SysML". *Machines* 10.10 (2022), p. 864.

[2] Andrew V Goldberg and Robert E Tarjan. "A new approach to the maximum-flow problem". *Journal of the ACM (JACM)* 35.4 (1988), pp. 921–940.

[3] J Cole Smith and Yongjia Song. "A survey of network interdiction models and algorithms". *European Journal of Operational Research* 283.3 (2020), pp. 797–811.