Diskrétní simulace za použití knihovny SimPy

Panovský Tomáš

29. října 2025

1 Úvod

V této bakalářské práci se zabývám diskrétní simulací za použití knihovny SimPy v Pythonu. Knihovna Simpy umožňuje modelovat procesy, jenž probíhají souběžně, a mohou být zastaveny, nebo pozastaveny na určitou dobu. Praktická část bakalářské práce obsahuje aplikaci simulující průběh hudebního festivalu. Cílem simulace je zjistit, jak se návštěvníci pohybují a kde vznikají fronty, což může pomoci při organizaci reálného festivalu.

2 Systém

str17 Reálnou skutečnost, kterou chceme analyzovat, označujeme jako systém. Analyzovat znamená sledovat, jak se různé části systému chovají v čase, a vyvozovat závěry o efektivitě, kapacitě nebo chování systému.

Na systémy můžeme nahlížet jako diskrétní nebo spojité, přičemž záleží na úhlu pohledu a vlastnostech, které v systému převažují. Diskrétní systém je takový, u kterého se stavové proměnné mění pouze v diskrétních časových okamžicích. Příkladem diskrétního systému je hudební festival, kde se stavové proměnné, například počet návštěvníků ve frontě u stánku s pivem, mění pouze tehdy, když návštěvník přijde na řadu, nebo dostane pivo a odejde. Oproti tomu spojitý systém je takový systém, u kterého se stavové proměnné mění plynule v čase, nikoli pouze v diskrétních okamžicích.

Stav systému je definován jako soubor stavových proměnných, které jsou nezbytné k popisu systému v libovolném okamžiku. V simulaci hudebního festivalu mohou být možné stavové proměnné například počet lidí čekajících ve frontách u stánků s občerstvením, počet lidí právě sledujících koncert, nebo čas příchodu dalšího návštěvníka do areálu.

2.1 Komponenty systému

Entita je objekt zájmu v systému. V našem případě mohou být entitami například návštěvníci festivalu, stánky s občerstvením nebo pódium.

Atribut je vlastnost entity, například počet lidí u stánku, nebo informace o tom, zda má návštěvník hlad nebo je unavený.

Aktivita představuje časově omezenou činnost entity, například čekání ve frontě, sledování koncertu nebo nákup jídla.

Událost (event) je okamžitá změna stavu systému. Události dělíme na Endogenní a Exogenní. Endogenní probíhají uvnitř systému a jsou způsobeny chováním jeho komponent, např. návštěvník dokončí konzumaci jídla a odchází od stánku. Exogenní probíhají v prostředí systému a ovlivňují ho zvenčí, například náhlý déšť.

3 Úvod do simulace

str6 Simulace je napodobení systému v čase. Cílem je vytvořit umělou historii stavu daného systému, a následně pozorovat tuto uměle vytvořenou historii za účelem vyvození závěrů o provozních vlastnostech systému, tedy o měřitelných charakteristikách a výkonnosti systému, jako je například délka front u stánků, průměrná doba čekání návštěvníků nebo hustota návštěvníků u pódia.

Chování systému vyvíjejícího se v čase se zkoumá pomocí simulačního modelu. V této práci je simulační model vytvořen v knihovně SimPy. Simulační model poté může být použit k prozkoumání široké škály otázek typu „co by se stalo, kdyby“ týkajících se reálného systému. Například můžeme zkoumat, zda je daný počet stánků s občerstvením dostatečný pro obsloužení všech návštěvníků festivalu bez dlouhých front.

Možné změny systému pak lze nejprve nasimulovat, aby bylo možné předpovědět jejich dopad na výkonnost systému. Simulace může být také použita ke studiu systémů ve fázi návrhu, ještě před jejich samotnou realizací.

potud snad cajk!!

4 Model systému

str. 13 Model je definován jako reprezentace systému za účelem jeho studia. Pro většinu studií je nutné zvažovat pouze ty aspekty systému, které ovlivňují problém,

který je předmětem zkoumání. Modely lze klasifikovat jako statické nebo dynamické, deterministické nebo stochastické a diskrétní nebo spojité. Statický simulační model reprezentuje systém v určitém časovém okamžiku, zatímco dynamické modely reprezentují systémy, jak se mění v čase. Simulační modely, které neobsahují náhodné vstupní proměnné, se klasifikují jako deterministické. Deterministické modely mají známou množinu vstupů, která vede k jednoznačné množině výstupů. Stochastický simulační model má jeden nebo více náhodných vstupů, které vedou k náhodným výstupům. Diskrétní a spojité modely jsou definovány obdobně jako u systémů.

Pro studium hudebního festivalu použijeme diskrétní, dynamický a stochastický model: diskrétní, protože stavové proměnné se mění pouze v konkrétních okamžicích; dynamický, protože sleduje vývoj systému v čase během celé doby trvání festivalu; a stochastický, protože některé vstupy – například časy příchodů návštěvníků, doba čekání u stánku nebo délka sledování koncertu – jsou náhodné.

5 Diskrétní simulace

str. 19 Simulace diskrétních systémů představuje modelování systémů, ve kterých se stavové proměnné mění pouze v diskrétní množině časových okamžiků. Tyto modely se neřeší analytickými, ale numerickými metodami. Analytické metody využívají deduktivní přístup matematiky k řešení modelu, například pomocí diferenciálního počtu lze stanovit optimální zásobovací politiku s minimálními náklady. Naproti tomu numerické metody používají výpočetní postupy, při nichž se modely nespočítají, ale simulují. Simulační modely reálných systémů bývají často rozsáhlé a vyžadují zpracování velkého množství dat, proto se tyto simulace obvykle provádějí s pomocí počítače.

6 Simulace v SimPy

SimPy je open-source knihovna pro jazyk Python, která slouží k modelování a simulaci diskrétních systémů událostí. Umožňuje popisovat systémy pomocí procesů, které představují jednotlivé entity nebo činnosti v systému, a které mezi sebou mohou interagovat prostřednictvím událostí. Každý proces je v SimPy implementován jako generátor (funkce využívající klíčové slovo `yield`), což umožňuje simulovat plynutí času a pozastavovat vykonávání procesů, dokud nenastane určitá událost.

Základním prvkem simulace je objekt `Environment`, který reprezentuje simulační prostředí. Tento objekt řídí běh celé simulace, spravuje plán událostí a posouvá simulační čas. Kdykoliv proces použije příkaz `yield`, předává řízení zpět prostředí `Environment`, které vyhodnotí, jaká událost má nastat jako další. Po uplynutí

naplánovaného času nebo splnění podmínky je proces znovu spuštěn od místa, kde byl pozastaven.

SimPy poskytuje také třídy `Resource`, `Container` a `Store`, které slouží k modelování omezených zdrojů, jako jsou fronty, zásoby nebo datové toky. Tyto objekty umožňují řídit interakce mezi procesy a simulovat situace, kdy více entit soutěží o stejný prostředek – například fronty návštěvníků čekajících u stánku s jídlem.

7 Příklad metody

Pro ilustraci uvádím metodu `go_to_festival_area()`, která simuluje průchod návštěvníka vstupním turniketem do areálu festivalu.

```
def go_to_festival_area(self, entrances):

    yield self.festival.timeout(random.expovariate(1/5))
    entrance_id = occupied.index(min(occupied))
    entrance = entrances[entrance_id]
    occupied[entrance_id] += 1

    with entrance.request() as req:
        queue_start = self.festival.now
        yield req

        queue_waiting_time = self.festival.now - queue_start
        entry_time = random.uniform(1, 3)
        yield self.festival.timeout(entry_time)

    self.state["location"] = resources.Location.FESTIVAL_AREA
    occupied[entrance_id] -= 1
```

Metoda `go_to_festival_area()` simuluje vstup návštěvníka jedním z dostupných vstupů. Metoda přijímá dva argumenty, a to instanci návštěvníka - self, a SimPy resource entrances. SimPy resource jsou objekty, ke kterým návštěvníci v simulaci přistupují, entrances je tedy seznam vstupů, mezi kterými si návštěvník může vybrat.

Každý návštěvník dorazí po náhodném zpoždění, které je generováno pomocí `random.expovariate(1/5)`, což modeluje příchody návštěvníků s průměrem 5 časových jednotek. Metoda využívá k simulování zastavení času pro návštěvníka funkci `yield self.festival.timeout()`, která na počet zadaných časových jednotek danou instanci uspí.

Návštěvník si vybere vstup, u kterého je aktuálně nejméně lidí čekajících ve frontě, což je sledováno pomocí globálního seznamu `occupied`. Následně požádá o přístup ke vstupu pomocí `entrance.request()`. Proměnná `queue_waiting_time` uchovává dobu, po kterou návštěvník čekal ve frontě, než se dostal „na řadu“. Po získání přístupu stráví návštěvník náhodnou dobu kontolou během vstupu, simulovanou pomocí `random.uniform(1, 3)` a `yield`

`self.festival.timeout(entry_time)`. Nakonec se počet lidí u daného vstupu sníží, což znamená, že návštěvník dokončil průchod vstupem, a nastaví se mu atribut „location“ na FESTIVAL_AREA.

Tento přístup umožňuje sledovat délku front a čekací doby u jednotlivých vstupů, což poskytuje užitečné informace pro analýzu průchodnosti a vytíženosti vstupů během festivalu.