

Инструкция пользователя

SAMPO – фреймворк с открытым исходным кодом для решения промышленных задач в части адаптивной оптимизации планирования. Области применения:

1. Одно- и многокритериальная оптимизация планирования бизнес-процессов в условиях ограничений.
2. Автоматизации выбора стратегии планирования с учетом корпоративного опыта на основе анализа ретроспективных данных.

На рисунке 1 приведена общая структура фреймворка.

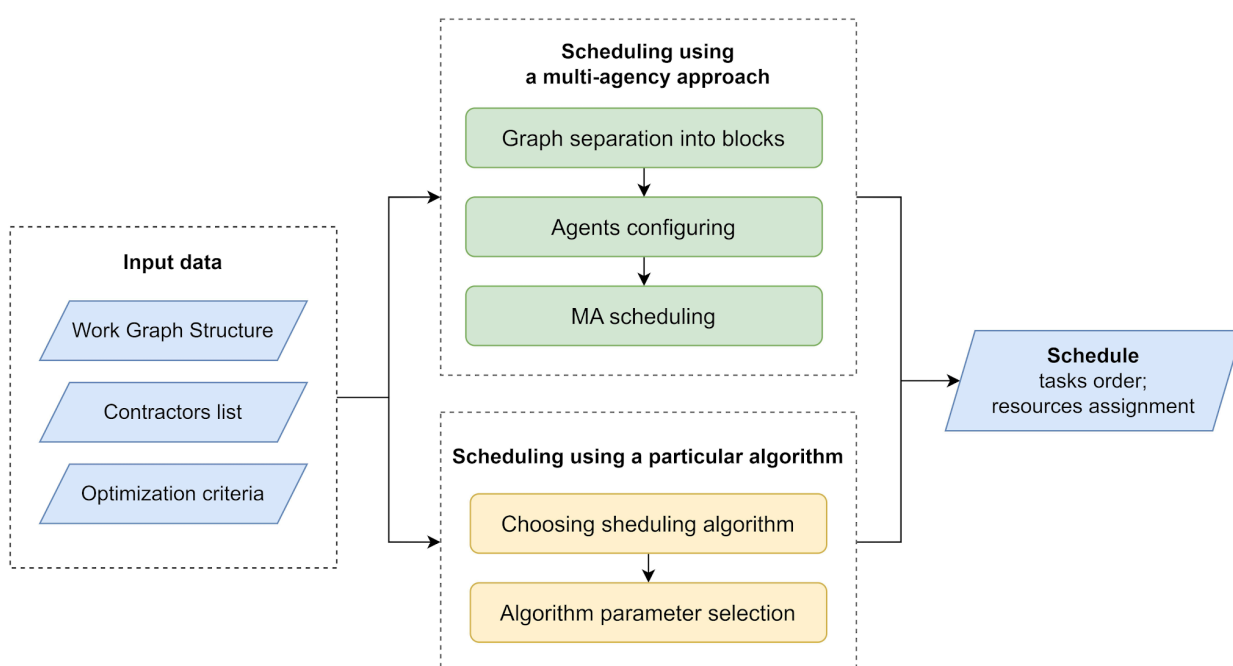


Рисунок 1 – Структура фреймворка, реализующего алгоритмы выбора стратегии и оптимизации планирования бизнес-процессов в условиях ограничений.

Для решения задачи оптимизации планирования бизнес-процесса на вход фреймворку необходимо передать информацию о графе задач с зависимостями между ними и ресурсными требованиями, информацию о выделенных на проект ресурсах, а также метрику(-и) оптимизации. На основе переданных данных в зависимости от выбранного алгоритма планирования планировщик строит оптимальную последовательность задач и распределение ресурсов в соответствии с заданной метрикой(-ами).

Для решения задачи автоматизации выбора стратегии планирования с

учетом корпоративного опыта на основе анализа ретроспективных данных на вход фреймворку необходимо передать структурированную информацию о реализованных расписаниях работ и назначениях ресурсов. На основе переданной информации идет определение оптимальной структуры зависимости задач в графе, назначении ресурсов и выбор алгоритма планирования с последующим построением расписания.

Пример работы с фреймворком для решения задачи оптимизации планирования бизнес-процессов в условиях ограничений

- **Шаг 1.** Инициализация экземпляров объектов WorkGraph и Contractor на основе входных данных

```
from sampo.schemas.graph import WorkGraph
from sampo.schemas.contractor import Contractor

wg = WorkGraph.load(input_folder_name, graph_file_name)
contractors = [Contractor.load(contractors_folder, contractor_info_file) for
contractor_info_file in contractors_folder]
```

- **Шаг 2.** Инициализация экземпляра класса Scheduler и передача ему ряда именованных аргументов

```
from sampo.scheduler.genetic import GeneticScheduler

scheduler = GeneticScheduler(number_of_generation=500,
                             mutate_order=0.9,
                             crossover_order=0.9,
                             mutate_resources=0.7,
                             crossover_resources=0.7,
                             size_of_population=100,
                             size_of_selection=500
)
```

Параметры конфигурации:

- o number_of_generation – количество поколений генетического алгоритма планирования;
- o mutate_order – вероятность мутации порядка работ в расписании;
- o crossover_order – вероятность кроссовера порядка работ в расписании;

- o `mutate_resources` – вероятность мутации назначения ресурсов на задачи в расписании;
- o `crossover_resources` – вероятность кроссовера назначения ресурсов на задачи в расписании;
- o `size_of_population` – количество особей в начальной популяции;
- o `size_of_selection` – количество особей в популяции после селекции.

Во фреймворке SAMPO доступно четыре класса планировщиков:

- o `HEFTScheduler`;
- o `HEFTBetweenScheduler`;
- o `TopologicalScheduler`;
- o `GeneticScheduler`.

Каждый из них имеет различные настраиваемые гиперпараметры, которые должны быть переданы при создании объекта планировщика.

- **Шаг 3.** Создание пайплайна планирования: передача входных данных, объекта планировщика и запуск процесса построения расписания

```
from sampo.pipeline import SchedulingPipeline

schedule = SchedulingPipeline.create() \
    .wg(wg) \
    .contractors(contractors) \
    .schedule(HEFTScheduler()) \
    .finish()
```

Компонент размещен в зеркалированных репозиториях:

- <https://github.com/aimclub/sampo>
- <https://gitlab.actcognitive.org/itmo-sai-code/sampo>