Формальная постановка задачи оптимизации расписания с использованием алгоритма имитации отжига

Кяжин Никита Олегович

23 октября 2024 г.

Формальная постановка задачи

Дано:

- Пусть $J = \{j_1, j_2, \ldots, j_N\}$ – множество заданий, где N – количество заданий.
- Пусть $\tau = \{t_1, t_2, \ldots, t_N\}$ – для каждого задания j_i задано время выполнения $t_i > 0$.
- Пусть $P = \{p_1, p_2, \ldots, p_M\}$ – множество процессоров, на которых выполняются задания, где M – количество процессоров.

Расписание:

Расписанием является булева матрица $S^{N \times M}$, в которой $s_{i,j} \in \{0, 1\}$, где i находится в диапазоне от 1 до N, а j – в диапазоне от 1 до M. Значение $s_{i,j} = 1$ означает, что задание i выполняется на процессоре j, а $s_{i,j} = 0$ – что задание i не выполняется на процессоре j.

Требуется:

Построить расписание S, при котором все задания J будут выполнены на множестве процессоров P без прерываний, с учетом ограниченных ресурсов, и не будет пересечений в использовании процессоров.

Минимизируемый критерий:

- В зависимости от остатка от деления на 2 контрольной суммы CRC32 от фамилии и инициалов выбирается один из следующих критериев:
  - **Критерий K_1 (разбалансированность расписания):**
    Разбалансированность расписания определяется как разность между максимальным и минимальным временем завершения заданий:
    $$K_1 = T_{\max} - T_{\min}, \tag{1}$$
    где $T_{\max} = \max_{i \in \{1,\ldots,m\}} c_i$ – максимальное время завершения задания на любом из процессоров, а $T_{\min} = \min_{i \in \{1,\ldots,m\}} c_i$ – минимальное время завершения задания на любом из процессоров.
    Необходимо минимизировать разбалансированность:
    $$S^* = \arg\min_S K_1. \tag{2}$$
  - **Критерий K_2 (суммарное время ожидания):**
    Суммарное время ожидания определяется как сумма моментов завершения всех заданий:
    $$K_2 = \sum_{i=1}^{n} c_i. \tag{3}$$
    Необходимо минимизировать суммарное время ожидания:
    $$S^* = \arg\min_S K_2. \tag{4}$$

Ограничения

- Для каждого процессора $P_j \in P$ в любой момент времени он может выполнять не более одного задания.
- Каждое задание $J_i \in J$ должно быть выполнено только один раз и только на одном процессоре.
- Времена начала выполнения заданий s_i должны быть выбраны так, чтобы не было конфликтов в использовании процессоров.