1 Постановка задачи

1.1 Задание по алгоритму имитации отжига

Автор: Балашов В.В.

Реализуйте алгоритм имитации отжига (обязательно: последовательный, опционально: параллельный). Решите с его помощью прикладную задачу из области построения расписаний. Выполните исследование производительности реализованного алгоритма.

1.2 Прикладная задача

Дано N независимых работ, для каждой работы задано время выполнения. Требуется построить расписание выполнения работ без прерываний на M процессорах. На расписании должно достигаться минимальное значение критерия.

- Критерий K1: длительность расписания, (т.е. время завершения последней работы)
- Критерий K2: суммарное время ожидания (т.е. сумма, по всем работам, времён завершения работ)
- Критерий K3: разбалансированность расписания (т.е. значение разности Tmax-Tmin, где Tmax - наибольшая, по всем процессорам, длительность расписания на процессоре; Tmin - аналогично, наименьшая длительность)

Выбор критерия: на основании остатка от деления на 3 контрольной суммы CRC32 от фамилии и инициалов студента (в формате "ИвановАБ").

Онлайн-расчёт CRC32: https://ru.functions-online.com/crc32.html

- остаток 0: критерий K3
- остаток 2: критерий K2
- остаток 1: критерий K1

Пояснение к критериям (показаны оси времени процессоров; слева номер процессора; выполняются работы А, Б, В; каждая буква - 1 единица времени выполнения соответствующей работы):

0: АААББ

1: ВВВВВВВВВВВ

Значение критерия К1: 11 (последним освобождается процессор 1)

Значение критерия К2: 19=3+5+11 (сумма времен завершения работ А, Б, В; эти времена считаются не от стартов работ, а от начала расписания)

Значение критерия К3: 6 (разность между временами освобождения процессоров 1 и 0)

1.3 Формальная постановка задачи

Необходимо описать формальную постановку задачи (ФПЗ).

Примеры формальных постановок задач планирования приведены в курсе В. Костенко.

Необходимые составляющие ФПЗ:

- ДАНО: (описание входных данных)
- ТРЕБУЕТСЯ: (описание результата работы алгоритма, т.е. расписания)
- МИНИМИЗИРУЕМЫЙ КРИТЕРИЙ: (описание минимизируемого критерия)

ФПЗ отличается от постановки на естественном языке тем, что для всех входных и выходных данных введены формальные обозначения, и в терминах этих обозначений сформулирован оптимизированный (минимизируемый) критерий.

ФПЗ должна быть описана в формате TeX, по которому должен быть сформирован PDF-файл.

1.4 Требования к реализации алгоритма ИО

Далее под "лекцией"имеется в виду лекция В. Костенко по алгоритмам ИО.

Требования к последовательной реализации:

- последовательная схема ИО должна быть реализована в виде набора классов C++
    - головной класс: реализует основной цикл ИО в виде, изолированном от конкретных деталей задачи за счёт использования перечисленных далее абстрактных

классов; схема работы основного цикла - в соответствии со слайдом "Общая схема"лекции

  - абстрактный класс для представления решения
  - абстрактный класс для операции изменения (мутации) решения
  - абстрактный класс для закона понижения температуры
  - три класса для трёх законов понижения температуры, описанных в лекции

- алгоритм ИО для конкретной задачи должен быть реализован за счет:
  - реализации класса для представления решения
  - реализации класса для операции мутации решения (содержание операции - на усмотрение студента)
  - выбора закона понижения температуры (одного из трех реализованных в виде классов)
  - "подстановки"этих конкретных классов в головной класс, реализующий схему ИО
- набор числовых параметров алгоритма (начальная температура и т.п.): в соответствии с содержимым лекции
- критерий останова (его можно хардкодить):
  - отсутствие улучшения решения в течение K=100 итераций
  - улучшение решения - это нахождение нового решения с меньшим значением минимизируемого критерия, чем у НАИЛУЧШЕГО ранее найденного решения
- не забывайте переинициализировать датчик случайных чисел, иначе алгоритм станет детерминированным!

Требования к параллельной реализации:

- необходимо реализовать параллельный алгоритм ИО с синхронизацией, работающий по описанной в лекции схеме со сбором лучших решений и рассылкой наилучшего

- параллельный алгоритм ИО должен быть реализован как набор параллельно работающих экземпляров последовательного алгоритма ИО (требования к реализации последовательного алгоритма указаны выше)

- критерий останова параллельного алгоритма: 10 итераций внешнего цикла (итерация = запуск набора экземпляров последовательного ИО) без улучшения решения

Критерий останова (его можно хардкодить):

- отсутствие улучшения решения в течение K=100 итераций

- улучшение решения - это нахождение нового решения с меньшим значением минимизируемого критерия, чем у НАИЛУЧШЕГО ранее найденного решения

Вариант 1: многопоточная реализация с обменом данными через разделяемую память, на основе pthreads или C++ threads (std:thread и т.п.)

Вариант 2: многопроцессная реализация с обменом данными через сокеты (локальные UNIX-сокеты PF_UNIX) или очереди сообщений POSIX.

Число потоков/процессов Nproc - числовой параметр реализации.

Выбор варианта параллельной реализации: на основании значения контрольной суммы CRC32 от фамилии и инициалов студента (в формате "ИвановАБ"). Если контрольная сумма нечётная, то вариант 1. Если чётная, то вариант 2.

1.5 Требования к исследованию

Написать генератор входных данных (описание набора работ). Формат данных - XML или CSV. Параметры генератора: число процессоров, число работ, диапазон длительностей работ.

1.6 Исследование последовательного алгоритма

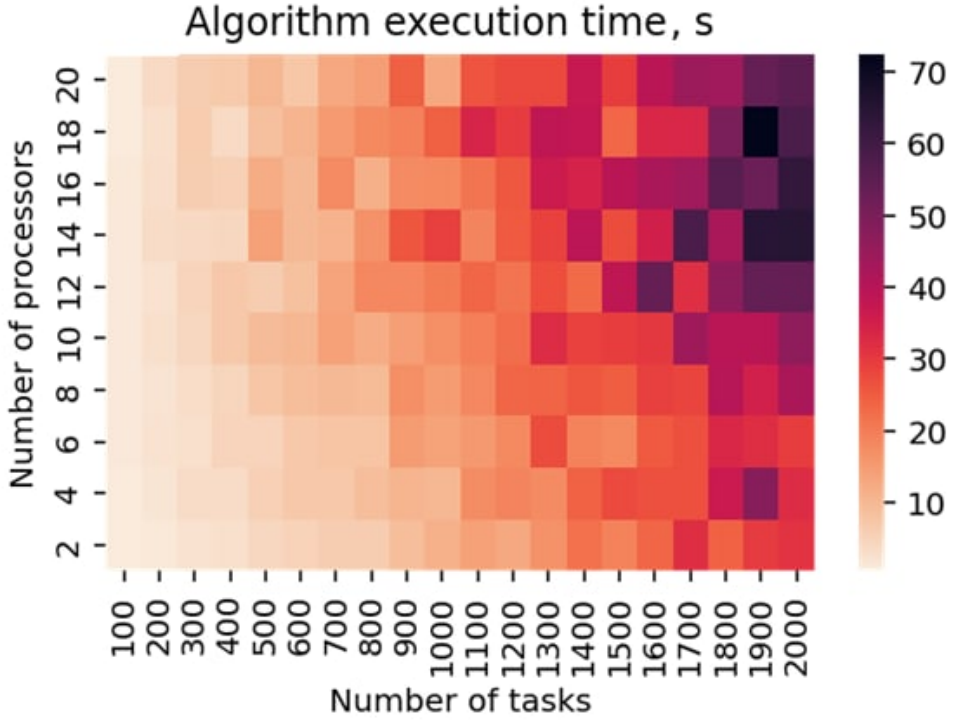
(это обязательно и в случае реализации параллельного алгоритма, установите в нем Nproc=1)

Экспериментально определите, при каких количествах процессоров и работ (N и M) последовательный алгоритм ИО работает больше 1 минуты хотя бы с одним из законов понижения температуры. С каким из законов понижения температуры на таких

алгоритм работает дольше всего на таких "тяжёлых"входных данных? Находит ли он при этом лучшие решения, чем при других законах понижения температуры?

В дальнейших экспериментах используйте один закон понижения температуры. Используйте данные, на которых последовательный ИО работает больше 1 минуты.

При исследовании последовательного алгоритма постройте "температурную карту"(heat map) зависимости среднего (по 5 прогонам на одних и тех же данных) времени работы алгоритма от значений M и N. Пример:



1.7 Исследование параллельного алгоритма

Экспериментально определите (постарайтесь найти компьютер с минимум 4 процессорными ядрами):

- какое значение Nproc следует задавать, чтобы параллельная реализация находила решение быстрее, чем последовательная. Или она будет работать точнее (находить лучшее решение), но не быстрее?

- повышение Nproc выше какого значения не дает значительного (более чем на 10%)

прироста по скорости. под приростом по скорости понимается сокращение длительности работы алгоритма (от старта до завершения)

При исследовании параллельного алгоритма постройте график зависимости времени работы параллельного алгоритма от значения Nproc (на фиксированном наборе входных данных).

1.8 Рекомендации по отладке алгоритма ИО

Отлаживайте с критериями-отрицаниями, тогда нахождение оптимального решения проверяется просто:

- отрицание К1 => строится самое длинное расписание (все работы на одном процессоре; если алгоритм хорошо рандомизирован, то от прогона к прогону процессор будет разным, а работы будут расположены в разном порядке)
- отрицание К3 => то же, что для отрицания К1
- отрицание К2 => то же, что для отрицания К1, плюс работы выстраиваются от самой длинной к самой короткой по убыванию длительности

1.9 Оценка составных частей задания в баллах

(обязательно) Формальная постановка задачи: 2 балла

(обязательно) Реализация и исследование последовательного алгоритма ИО: 4 балла

(опционально) Реализация и исследование параллельного алгоритма ИО: 4 балла

Если реализован параллельный алгоритм ИО, то за "реализацию последовательного алгоритма"засчитывается работоспособный (!) при прогоне в однопроцессном/однопоточном режиме параллельный алгоритм ИО.

Реализация без исследования оценке не подлежит (0 баллов).