# Исследование последовательного алгоритма

* 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?

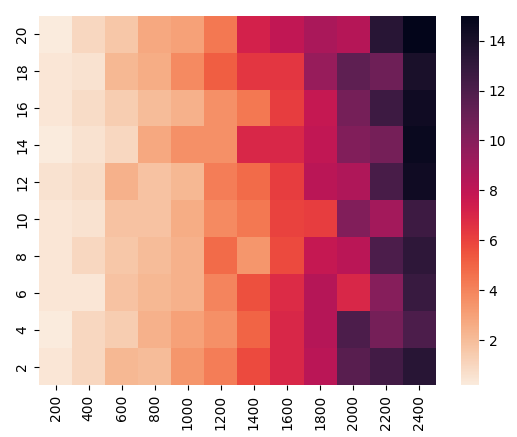
Эксперимент проводился следующим образом: n=5500, m=10, n\_proc=1, для каждого из правил понижения температуры производился запуск алгоритма на одинаковых входных данных. По результатам эксперимента были построены следующие таблицы:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Номер измерения | Правило Больцмана | | Правило Коши | | Смешанное правило | |
| Результат | Время | Результат | Время | Результат | Время |
| 1 | 43485879 | 44 | 43485851 | 51 | 43486016 | 47 |
| 2 | 51581861 | 50 | 51581271 | 60 | 51580928 | 57 |
| 3 | 50182112 | 60 | 50183040 | 44 | 50182126 | 56 |
| 4 | 52244441 | 47 | 52243660 | 64 | 52243729 | 64 |
| 5 | 52121363 | 47 | 52121633 | 49 | 52121587 | 50 |
| 6 | 50832826 | 51 | 50832910 | 57 | 50832591 | 66 |
| 7 | 50731021 | 57 | 50732052 | 43 | 50730881 | 79 |
| 8 | 52187174 | 48 | 52186284 | 71 | 52187004 | 48 |

Быстрее, как правило, отрабатывает алгоритм с понижением температуры по правилу Больцмана. Результат у всех правил примерно одинаков на любых входных данных.

Нельзя сказать, что какой-то закон работает дольше, чем остальные.

* 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.



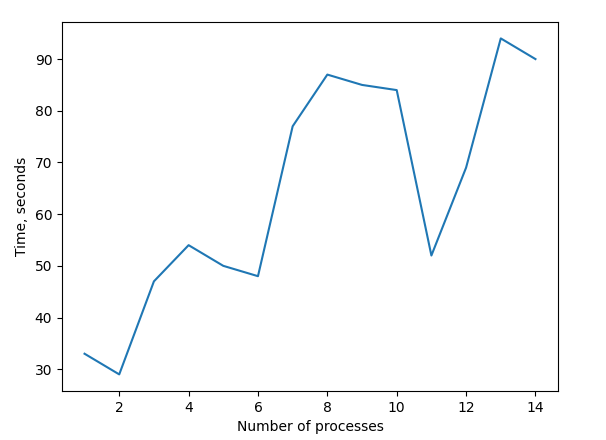
# Исследование параллельного алгоритма

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

Было проведено экспериментальное исследование на компьютере с 6-ти ядерным процессором с n=4000, m=10:

|  |  |  |
| --- | --- | --- |
| Число процессов | Результат программы | Время работы |
| 1 | 26829382 | 33 |
| 2 | 26829183 | 29 |
| 3 | 26829093 | 47 |
| 4 | 26828666 | 54 |
| 5 | 26828946 | 50 |
| 6 | 26828906 | 48 |
| 7 | 26828507 | 77 |
| 8 | 26828386 | 87 |
| 9 | 26828336 | 85 |
| 10 | 26828734 | 84 |
| 11 | 26828902 | 52 |
| 12 | 26828574 | 69 |
| 13 | 26828602 | 94 |
| 14 | 26828349 | 90 |

Сначала с ростом числа процессов решение улучшается, однако и время его работы увеличивается. Начиная с n\_proc=10 результат колеблется, то улучшаясь, то ухудшаясь, время работы зависит от результата – чем он лучше, тем больше время работы.



* Повышение Nproc выше какого значения не дает значительного (более чем на 10%) прироста по скорости?

У прироста по скорости не наблюдается зависимости от числа процессов. Только в самом начале, программа с n\_proc=2 работает незначительно быстрее, чем для n\_proc=1 (все же больше, чем на 10%), однако потом время работы становится больше, однако принося лучшие результаты.