УДК 004.89

Д.Е. Жаравин
Вологодский государственный университет

ВЛИЯНИЕ ПАРАЛЛЕЛЬНЫХ ВЫЧИСЛЕНИЙ НИЗКОГО УРОВНЯ НА СКОРОСТЬ РАБОТЫ ГЕНЕТИЧЕСКИХ АЛГОРИТМОВ

В статье представлены результаты исследования скорости работы базовых методик, составляющих большую часть работы генетических алгоритмов – создание и изменение числовых матриц. Проверена скорость работы алгоритмов, созданных при помощи двух различных приемов, отличающихся способом организации данных в памяти компьютера. Определены важные аспекты, влияющие на скорость работы генетических алгоритмов, которые можно использовать в дальнейших исследованиях.

Структуры данных, параллельные вычисления, генетические алгоритмы, задачи оптимизации, поиск решения.

Во многих сферах деятельности одной из основных проблем реализации алгоритмов для решения оптимизационных задач является скорость обработки большого количества данных.

Стандартные математические методы оптимизации той или иной целевой функции сводятся к перебору значений большого пространства поиска, что значительно увеличивает время работы алгоритмов. Поэтому с начала 1980-х годов, когда произошло резкое развитие микропроцессоров, создаются другие методы оптимизации, получившие название «метаэвристические» [1].

Понятие метаэвристики в программировании включает в себя большое количество разнообразных подходов и алгоритмов, однако все они обладают одной ключевой идеей – моделирование процессов биологической эволюции с целью нахождения наиболее подходящих решений.

Одно из направлений метаэвристики занимается так называемыми эволюционными вычислениями, в их состав входят генетические алгоритмы, функция которых связана с моделированием работы естественного отбора. Данная концепция строится на принципах наследования и передачи признаков от родителей к их потомкам, с последующей мутацией генов и отбором наиболее приспособленных особей в популяции.

Значительная часть современных технологий (в своей основе) была полностью или частично вдохновлена природными явлениями, а затем усовершенствована и успешно использована для создания новых решений [2]. Генетические алгоритмы не стали исключением. Принципы их работы действительно позволяют находить решение сложных задач, сферы влияния которых сегодня выходят далеко за рамки простой математики.

Для определения эффективности того или иного алгоритма принято оценивать его работу по двум основным техническим параметрам: скорости и устойчивости [3].

Под устойчивостью работы алгоритма имеется в виду его возможность наиболее эффективно и быстро адаптироваться к изменениям пространства поиска, а также обрабатывать неполные объемы входящих данных. Еще одной очень важной оценкой устойчивости является способность программы находить выход из точек локальных экстремумов целевой функции, и чем быстрее алгоритм это сделает, тем устойчивее считается его работа.

Как показывает практика, для большого количества алгоритмов попадание в точки локальных максимумов и минимумов является одной из самых серьезных проблем. Для отработки таких ситуаций существуют специальные функции, на которых разработчики тестируют свои программные продукты. В качестве примера (рис. 1) представлена функция двух переменных, полученная путем преобразования и масштабирования Гауссовых распределений, с ее помощью можно выявлять подобные ошибки и отлаживать программу.

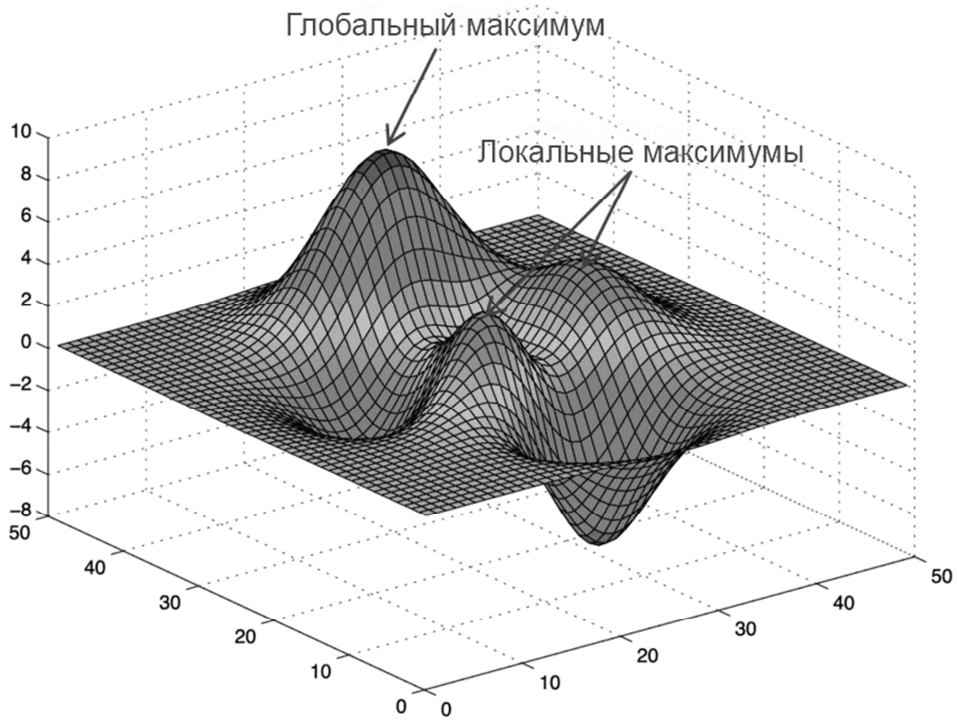

Рис. 1. График функции $z = 3(1-x)^2 e^{-x^2-(y+1)^2} - 10\left(\frac{x}{5}-x^3-y^5\right)e^{-x^2-y^2} - \frac{1}{3}e^{-(x+1)^2-y^2}$

Критерий устойчивости работы эволюционных методов является одной из самых обширных тем и требует отдельного исследования в дальнейшем.

Вторая важная характеристика – скорость работы алгоритма – показывает, за какое время он достигает оптимального или наиболее приближенного решения поставленной задачи. Скорость работы может зависеть он нескольких факторов, к ним относятся: структуры данных, которые применяются в расчетах, способ и синхронизация многопоточных вычислений, а также реализация самого алгоритма.

Использование параллельных вычислений является важным критерием для обеспечения высокой скорости работы алгоритмов, так как именно одновременная обработка потенциальных решений способна сократить время поиска оптимального решения, особенно когда объем обрабатываемых данных действительно велик.

В качестве примера использования многопоточных вычислений можно привести библиотеку NumPy, которая разработана специально для научных вычислений на языке программирования Python. Отличительной особенностью этой библиотеки является также внутренняя реализация, которая позволяет эффективно распределять вычислительную нагрузку за счет разделения процессов, что достигается работой ее алгоритмов на более низком уровне абстракции.

При использовании NumPy значительный прирост производительности можно выявить даже на самых простых задачах, таких как создание числовой матрицы и суммирование ее элементов. Следовательно, при дальнейшем усложнении программы, разница в скорости работы будет прослеживаться еще более отчетливо.

В качестве примера для оценки производительности двух одинаковых алгоритмов, различающихся только в используемых структурах данных, были созданы две квадратные матрицы: первая создана при помощи стандартного списка в Python, а вторая – с использованием библиотеки NumPy. В ходе работы программы требуется сравнить время, которое необходимо затратить на создание матриц обоих типов.

В целом операции с матрицами данных занимают достаточно большую часть работы метаэвристических алгоритмов, поэтому в исследовании, помимо скорости создания, необходимо сравнить время, за которое будет произведено полное суммирование (операция может быть любой) всех элементов матриц.

Результаты обоих экспериментов представлены ниже (рис. 2), на вертикальной шкале отображено время, затраченное на создание и суммирование элементов матриц для каждого из исследуемых подходов: List и NumPy.

По данной диаграмме видно значительную разницу во времени даже при выполнении самых простых алгоритмов, работа которых занимает не более секунды, однако при увеличении объемов обрабатываемых данных, а также при усложнении операций над ними различие в производительности становится еще более заметно. При усложнении структуры, время работы алгоритма может увеличиваться в разы, занимая при этом от нескольких секунд, до нескольких минут и даже часов.

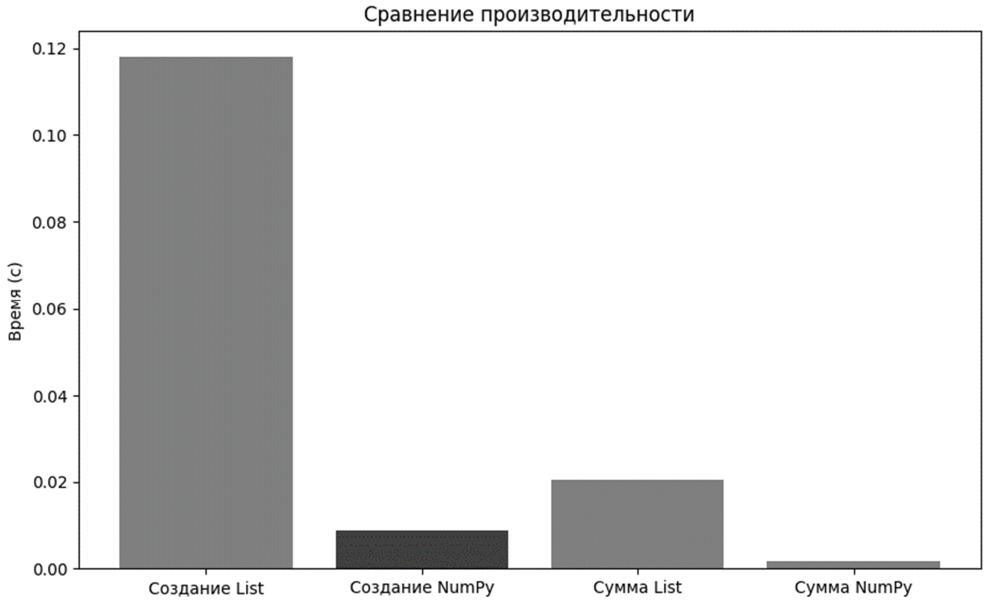

Рис. 2. Сравнение производительности работы двух алгоритмов

Таким образом, грамотное использование ресурсов, структур данных и методик при проектировании алгоритмов имеет большое значение в скорости выполнения любой программы, а в случаях с обработкой больших объемов данных эти критерии играют ключевую роль. Именно благодаря поддержке многопоточных вычислений, а также более оптимальной организации данных в памяти компьютера, методы NumPy позволяют достичь гораздо более высокой производительности по сравнению с аналогичными операциями, которые реализованы непосредственно на самом языке Python.

Как показывает практика, многопоточность и параллельные вычисления действительно занимают важное место в обработке больших объемов данных, в работе сложных алгоритмов и выполнении нестандартных задач. Но, несмотря на это, применение многопоточности на уровне языка Python (высокий уровень абстракции), может быть далеко не всегда оправдано, и эта тема требует намного более углубленного исследования в дальнейшем.

Именно поэтому, несмотря на то что генетические алгоритмы (в своем классическом понимании) появились достаточно давно, сегодня ведутся постоянные исследования во многих сферах и областях знаний, чтобы открыть новые подходы и методы в реализации подобных алгоритмов и систем. Иными словами, концепция разработки высококачественных решений сложных проблем и способность в дальнейшем адаптировать эти решения в условиях меняющейся среды была и остается актуальной на сегодняшний день.

Результаты данного исследования планируется использовать в дальнейшей работе по выявлению и настройке других параметров эволюционных вычислений, таких как устойчивость работы, многопоточность и т.д., которые, при правильном подходе, позволяют сократить время выполнения программы, что при работе с большими выборками данных может оказаться существенным преимуществом перед другими подобными системами.

Литература

1. Mitchell, M. An Introduction to Genetic Algorithms / M. Mitchell // The MIT Press. – 1998. – P. 162.

2. Жаравин, Д. Е. Исследование глобальных параметров генетических алгоритмов / Д. Е. Жаравин // Молодые исследователи – регионам : материалы Международной научной конференции (Вологда, 17 апреля 2023 г.) : / Министерство науки и высшего образования Российской Федерации и др. ; [главный редактор Л. О. Кочешкова]. – Вологда : ВоГУ, 2023. – 100 с.

3. Черноруцкий, И. Г. Методы оптимизации в теории управления / И. Г. Черноруцкий. – Санкт-Петербург : Питер, 2004. – 256 с.

D.E. Zharavin
Vologda State University

IMPACT OF LOW-LEVEL PARALLEL COMPUTING ON GENETIC ALGORITHMS SPEED

The article presents the results of research of the speed of the basic techniques that make up most of the work of genetic algorithms – the creation and modification of numerical matrices. The speed of work of algorithms is tested using two different techniques that differ in the way of organizing data in computer memory. Important aspects affecting the speed of genetic algorithms, which can be used in further research, are identified,

Data structures, parallel computing, genetic algorithms, optimization problems, solution finding.