DOI 10.37882/2223–2966.2020.07.20

ВЫБОР ЯЗЫКА ПРОГРАММИРОВАНИЯ ДЛЯ ПАРАЛЛЕЛЬНОЙ РЕАЛИЗАЦИИ АЛГОРИТМОВ И ПРОГРАММНЫХ СРЕДСТВ ОПТИМАЛЬНОГО ПАРАМЕТРИЧЕСКОГО СИНТЕЗА

CHOOSING A PROGRAMMING LANGUAGE FOR PARALLEL IMPLEMENTATION OF ALGORITHMS AND SOFTWARE TOOLS FOR OPTIMAL PARAMETRIC SYNTHESIS

A. Lagunova

Summary. This article is devoted to the choice of a programming language for the development of effective parallel algorithms and software tools for optimal parametric synthesis, as well as to the analysis of the relevance of parallel computing. The main methods of this research are a literary review of domestic and foreign scientific works, comparison of programming languages, and selection of the most optimal variant using the hierarchy analysis method. In the course of this work, the main advantages and disadvantages of several of the most commonly used programming languages are identified, and the choice of Python as a priority language is justified.

Keywords: high-level language, parallel programming, hierarchy analysis, information technology, Python, Java, C++.

Лагунова Алина Дмитриевна
schimka_06@mail.ru
Аспирант, Институт автоматики и процессов управления ДВО РАН, г. Владивосток

Аннотация. Данная статья посвящена выбору языка программирования для разработки эффективных параллельных алгоритмов и программных средств оптимального параметрического синтеза. Основными методами данного исследования является литературный обзор отечественных и зарубежных научных работ, сравнение языков программирования и выбор наиболее оптимального варианта с помощью метода анализа иерархий. В ходе данной работы выявлены основные преимущества и недостатки нескольких наиболее часто использующихся языков программирования, обоснован выбор языка Python в качестве приоритетного, а также произведен разбор актуальности параллельных вычислений на языке Python.

Ключевые слова: язык высокого уровня, параллельное программирование, анализ иерархий, информационные технологии, Python, Java, C++.

В современном мире все более быстрыми и интенсивными темпами развиваются цифровые и информационные технологии. Практически во всех сферах внедряются абсолютно новые, а также усовершенствуются прежние технические аппараты и средства. Программирование является одной из основных сфер деятельности большинства современных организаций и предприятий [9, 7,10]. Основная часть работы программистов непосредственно связана с написанием исходного кода на различных языках программирования. Каждый язык имеет свои особенности, достоинства и недостатки, поэтому крайне важно правильно выбрать язык программирования для решения конкретной задачи.

На сегодняшний день нет общепринятой систематической таксономии языков программирования. Тем не менее, существует множество черт, по которым можно провести классификацию: поколение; высокий или низкий уровень языка; парадигма программирования и т.д. Для решения задачи разработки эффективных параллельных алгоритмов и программных средств оптимального параметрического синтеза наиболее логично выбрать мультипарадигмальный язык высокого уровня 5-го поколения. Наиболее популярными языками, удовлетворяющими требованиям, согласно IEEE Spectrum [19], являются Python, Java и C++. Рассмотрим краткую характеристику данных языков.

Python. Является альтернативой стандартным вычислительным и математическим пакетам (Mathematica, Octave, MatLab и пр.), однако имеет привычную семантику языка и большое число библиотек. Из всех интерпретируемых языков Python выделяется большим и активным сообществом научных расчетов. Особенно широкое распространение данный язык получил в связи с появлением улучшенных библиотек, что сделало его серьезным конкурентом в решении задач манипулирования данными. Является мультипарадигмальным (3 парадигмы: функциональная, процедурная, объектно-ориентированная). По типу обрабатываемых данных — вычислительный. Распространяется бесплатно.

C++. Предоставляет возможности функционального и объектно-ориентированного программирования,

Таблица 1. Матрица парных сравнений критериев отбора

	Науч. вычисления	Кроссплат-форменность	Параллельные вычисления	Скорость	Лицензия	Доп. функции	Простота освоения
Научн. вычисления	1	4	4	5	5	7	7
Кроссплат-форменность	1/4	1	1	3	4	5	7
Параллельные вычисления	1/4	1	1	2	3	5	6
Скорость	1/5	1/3	1/2	1	2	4	4
Лицензия	1/5	1/4	1/3	2	1	4	4
Доп.функции	1/7	1/5	1/5	1/4	1/4	1	1/2
Простота усвоения	1/7	1/7	1/7	1/4	1/4	2	1

Таблица 2. Матрицы парных сравнений для каждого решения

	Научные вычисления			Кроссплатформенность			Параллельные вычисления		
	Python	C++	Java	Python	C++	Java	Python	C++	Java
Python	1	4	7	1	4	1	1	1/4	1/7
C++	1/4	1	2	1/4	1	1/4	4	1	1/3
Java	1/7	1/2	1	1	4	1	7	3	1

	Скорость			Лицензия			Доп.функционал			Простота освоения		
	Python	C++	Java	Python	C++	Java	Python	C++	Java	Python	C++	Java
Python	1	1/5	1/2	1	8	1	1	1/2	1/2	1	1/4	1/3
C++	5	1	1/3	1/8	1	1/8	2	1	1	4	1	1/3
Java	2	3	1	1	8	1	2	1	1	3	3	1

не потеряв при этом способность низкоуровневого взаимодействия с аппаратным обеспечением. За счёт чего реализуется производительность и гибкость при создании ПО. Высокий порог освоения языка за счёт сложной спецификации и необходимости самостоятельного контроля за ресурсами при выполнении программы. Является мультипарадигмальным (3 парадигмы: функциональная, процедурная, объектно-ориентированная). По типу обрабатываемых данных — символьный. Не является бесплатным.

Java. Подходит для большинства задач программирования. Уникальная адаптивность и универсальность Java сделали его языком программирования для многих компаний-разработчиков программного обеспечения по всему миру. Гибкая система безопасности. Является мультипарадигмальным (5 парадигм: объектно-ориентированная, обобщенная, процедурная, аспектно-ориентированная, конкурентная). Распространяется бесплатно.

Для определения наиболее подходящего языка воспользуемся методом анализа иерархий [8]. Основная задача, в рамках которой решается проблема выбора языка программирования, сформулирована следующим образом: «Разработка эффективных параллельных алгоритмов и программных средств оптимального параметрического синтеза». Для принятия решения, сформулируем критерии выбора, соответствующие цели поставленной задачи:

- Поддержка основных операций для научных вычислений.
- Кроссплатформенность
- Реализация параллельных вычислений
- Скорость работы
- Стоимость/лицензия
- Расширенный функционал в виде дополнительных функций обработки и отображения
- Простота освоения языка

Следует оговориться, что описываемые в рамках данной статьи критерии и условия отбора являются частным случаем и могут применяться в общем случае только с некоторыми оговорками.

Матрица парных сравнений для критериев отбора выглядит следующим образом (Таблица. 1). Высчитаем наибольшее собственное число (λ_{max}), нормализованный собственный вектор (W), индекс согласованности (ИС) и уровень общей связности (ОС). Усредненный индекс согласованности M для матриц 7x7, согласно [8], равен 1.32.

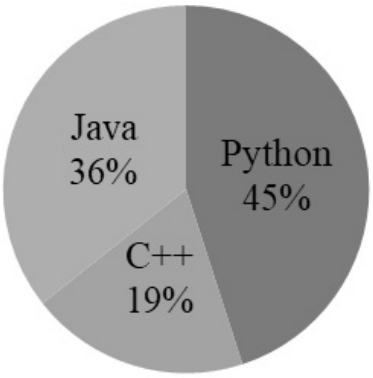

Рис. 1. Соотношение значений приоритета выбора языков

$$\lambda_{max} = 7.54;\ W = \begin{pmatrix} 0.4083 \\ 0.1876 \\ 0.1646 \\ 0.0979 \\ 0.0757 \\ 0.0304 \\ 0.0353 \end{pmatrix};$$

$$\text{ИС} = \frac{7.544-7}{7-1} = 0.0072;\ \text{ОС} = \frac{0.0072}{M} = 0.055$$

Согласно [8], уровень общей связности (ОС) не должен быть более 10%, чтобы данные могли считаться согласованными и им можно было доверять. В нашем случае, уровень общей связности — 5,5%, что является более чем приемлемым показателем и говорит о высокой достоверности данных.

Для составления матриц парных сравнений для каждого из рассматриваемых решений была использована информация об основных характеристиках языков и сравнение их по различным характеристикам, представленные в работах [16,17,18,12,14, 11]. Оценка альтернатив на основе полученной информации представлена в виде матриц парных сравнений (см. Таблица 2). Для каждой матрицы были высчитаны: максимальное собственное значение, собственный вектор, индекс согласованности и уровень общей связности. Вектор общих уровней связности:

$$\overline{OC} = (0.015, 0, 0.027, 0.002, 0, 0, 0.086)$$

Стоит отметить, что все значения уровней связности получились менее 10%, а средняя связность модели равна 2.3%, что говорит о высоком уровне согласованности данных, т.е. полученным результатам абсолютно точно можно доверять.

Осуществляя иерархический синтез, получаем следующий вектор решений: R = (0.4470, 0.1926, 0.3540)

Соотношение высчитанных значений приоритета выбора представлено на Рис. 1. Таким образом, опираясь на все полученные результаты, сделан выбор в пользу языка Python.

Параллельные вычисления в Python

Согласно данным, представленным в таблице 2, можно выделить критерий возможности и доступности параллельных вычислений как наиболее слабое звено Python. Разберем этот момент несколько подробнее и докажем, что хотя Python и проигрывает Java и C++ по этому параметру, тем не менее, параллельное программирование в Python показывает достаточно хорошие результаты в увеличении производительности и скорости работы программы, особенно в целях разработки алгоритмов оптимального параметрического синтеза.

Актуальность параллельного программирования связана с увеличением спроса на компьютеры, работающие с все более высокой скоростью. Примерами данных заказчиков являются фармацевты, ежедневно разрабатывающие лекарственные препараты посредством компьютерного тестирования. Астрономы, пытающиеся восстановить историю Вселенной, начиная с большого взрыва и заканчивая сегодняшним днем. Разработчики летательных аппаратов и БПЛА смогли бы получать более точные результаты при проведении тестирования своих комплексов и установок, не создавая при этом огромные полигоны и проводя экономически-невыгодные исследования. Однако, не смотря на рост мощности современного компьютера, его мощности никогда не хватит для проведения расчетов с колоссальным количеством данных, используемых в научно-исследовательских работах. На сегодняшний день в различных сферах деятельности существует множество задач со сложным нелинейным характером, оптимальное решение которых достаточно трудно-вычислимо с помощью классических

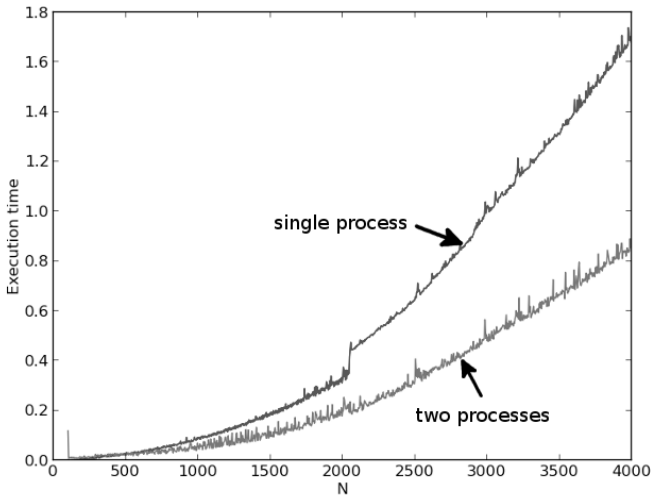

Рис. 2. Время работы однопоточной и двупоточной программы

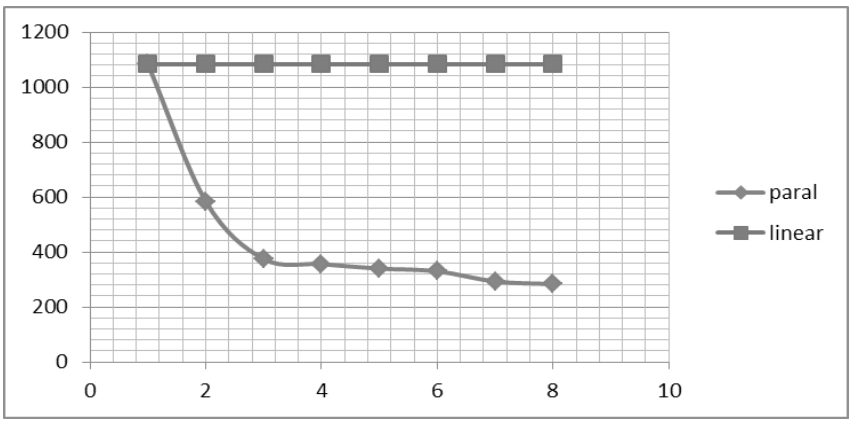

Рис. 3. Время работы (сек.) линейного и параллельного алгоритма для сетки 11 млн. элементов на 4х-ядерном ПК с количеством потоков от 1 до 8

методов вариационного и дифференциального исчисления средствами обычных последовательных ЭВМ. Одним из способов решения подобных задач является параллельное программирование [1,2].

С появлением многоядерных процессоров стала общеупотребительной практика распространять нагрузку на все доступные ядра. Существует два основных подхода в распределении нагрузки: использование процессов и потоков [3]. Использование нескольких процессов фактически означает использование нескольких программ, которые выполняются независимо друг от друга. Программированием это решается с помощью системных вызовов exec и fork. Такой подход создает большие неудобства в управлении обмена данными между этими программами. В качестве альтернативы существует другой подход — создание многопоточных программ. Обмен данными между потоками существенно упрощается. Но управление такими программами усложняется, и вся ответственность ложится на программиста.

На данный момент Python имеет несколько библиотек и функций для параллельных вычислений. Преследуя цель выполнения приложением нескольких задач одновременно, возможно использование потоков (threads) или процессов (processes) [13,15]. Данные методы позволяют приложением производить в одну и ту же единицу времени множество задач.

В Python есть стандартная библиотека subprocess, которая упрощает управление другими программами, передавая им опции командной строки и организуя обмен данными через каналы (pipe). Посмотрим на график ускорения расчетов с помощью subprocess в задаче на-

хождения функции, зависящей от расстояния между парами N точек в трехмерном пространстве (см. Ри.2)

В работе [6] решение оптимизационной задачи методом сеток сводится к распараллеливанию программы на несколько потоков посредством библиотеки multiprocessing. Данная библиотека в неявном для программиста виде распределяет потоки по ядрам. Результаты работы алгоритма для разного количества потоков представлены на рисунке 3.

Из графика видно, что лучшие результаты дало распараллеливание на 8 потоков — 283 сек, что в 3,82 раза быстрее линейных расчетов (1083 сек). Однако в многопоточных программах усложняется контроль за обменом данных между потоками. Глобальные ресурсы требуется остерегать от одновременного доступа со стороны нескольких потоков, с целью сохранности их целостности. Следовательно, наиболее актуально использование процессов, а не потоков.

Одной из наиболее актуальных с целью реализации параллельных вычислений для языка Python является библиотека MPI (mpi4py), в явном виде работающая с каждым ядром и процессом. Использование данной библиотеки для решения оптимизационных задач с помощью стохастических алгоритмов GWO и BA было представлено в работах [4,5]. Использование параллельных вычислений позволило значительно увеличить точность расчетов и скорость работы программы.

Кроме рассмотренных выше существует и другие библиотеки для параллельных вычислений, однако на данный момент они не являются широко используемыми в силу некоторых своих особенностей. Тем не менее, как уже говорилось выше, Python имеет огромную и регулярно пополняющуюся базу различных библиотек, функционал которых постоянно расширяется и улучшается. Более того, указанные выше данные о проигрыше в скорости вычислений по сравнению с другими языками являются усредненной оценкой для общих тестовых задач. Тем не менее, в зависимости от поставленной задачи различие в скорости может оказаться абсолютно незначительным. Таким образом мы потенциально имеем большое количество эффективных способов для параллельной реализации вычислений.

Заключение

В современном мире программирование является важной частью практически в любых сферах деятельности большинства современных организаций и предприятий, поэтому выбор языка программирования для разработки ПО является очень важным вопросом. Однако в зависимости от поставленной задачи критерии отбора приоритетного языка могут значительно отличаться.

В данной работе были рассмотрены языки программирования Python, Java и C++, а также произведен выбор языка для разработки эффективных параллельных алгоритмов и программных средств оптимального параметрического синтеза. Согласно результатам, полученным с помощью метода анализа иерархий, для данной задачи наиболее оптимальным языком разработки является Python. Поскольку параллельные вычисления в Python являются самым слабым критерием отбора (по сравнению с C++ и Java), также были подробно рассмотрены непосредственно сами способы их реализации и применение для решения оптимизационных задач. Согласно выше приведенным данным, параллельная реализация алгоритмов решения поставленной задачи на языке Python позволяет значительно увеличить скорость работы и точность расчетов, удовлетворяя при этом всем остальным требованиям. Проигрыш в скорости работы другим языкам при этом является незначительным.

ЛИТЕРАТУРА

1. Абрамов О.В., Катуева Я. В. Технология параллельных вычислений в задачах анализа и оптимизации / О. В. Абрамов, Я. В. Катуева // Проблемы управления.— 2003. № 4.— С. 11–15
2. Диго Г.Б., Диго Н. Б., Катуева Я. В. Применение детерминированных критериев в задачах стохастической оптимизации / Г. Б. Диго, Н. Б. Диго, Я. В. Катуева // Многопроцессорные вычислительные системы.— 2006.— № 2(12). С. 82–88
3. Копылов М.С., Бирюков Е. Д. Использование многопоточности в сценариях, написанных на языке Python, в системах автоматизированного проектирования трёхмерной графики // Новые информационные технологии в автоматизированных системах. 2018.
4. Лагунова А. Д. Алгоритм N-стай серых волков (GWO(N)) / А. Д. Лагунова // Международный академический вестник.— 2020.— № 1(45).— С. 48–59
5. Лагунова А. Д. Параллельные вычисления как способ повышения эффективности алгоритма летучих мышей (BA) / А. Д. Лагунова // Сборник статей IX Международной научно-практической конференции «Инновационное развитие науки и образования».— 2020.— С. 78–87
6. Лагунова А.Д., Назаров Д. А. Параллельный алгоритм решения задачи оптимального параметрического синтеза на основе метода сеток // Труды Международного симпозиума «Надежность и качество».— 2018.— Т. 1.— С. 255–258
7. Маркин А.С. применение языков программирования на предприятии / А.С, Маркин // VIA SCIENTIARUM — Дорога знаний.— 2019.— № 4.— С. 17–22
8. Саати Т. Принятие решений методом анализа иерархий // М.: Радио-Связь.— 1994–278 с.

9. Чернецов А. М. Возможности параллельного программирования в математических пакетах // Программные продукты и системы. 2016.
10. Фишер Й., Машков В. А., Литвиненко В. И. Применение языка программирования Python для решения задач самодиагностики на системном уровне / Й. Фишер, В. А. Машков, В. И. Литвиненко // Электротехнические и компьютерные системы.— 2015.— № 17 (93).— С. 48–54
11. Шкарбан А. С. Выбор языка программирования для задачи анализа данных методом анализа иерархий [Электронный ресурс] // Nauka-rastudent.ru.— 2016.— No. 02 (26). URL: http://nauka-rastudent.ru/26/3208/ (дата обращения: 24.06.2020)
12. Alzahrani N., Vahid F., Edgcomb A., Nguyen K., Lysecky R. Python Versus C++: An Analysis of Student Struggle on Small Coding Exercises in Introductory Programming Courses / N. Alzahrani, F. Vahid, A. Edgcomb, K. Nguyen, R. Lysecky // SIGCSE: Proceedings of the 49th ACM Technical Symposium on Computer Scienc.— 2018.— P. 86–91
13. Andrews, G. R. Fundamentals of multithreaded, parallel and distributed programming, Moscow: Williams, 2003.
14. Ezenwoye O. What Language? The Choice of an Introductory Programming Language [Электронный ресурс] / O. Ezenwoye // IEEE Frontiers in Education Conference (FIE).— 2018. URL: https://ieeexplore.ieee.org/abstract/document/8658592 (дата обращения: 24.06.2020)
15. Voevodin V. V. Parallel computing. Saint Petersburg: BHV-Petersburg, 2002.
16. About C++ [Электронный ресурс]. URL: https://isocpp.org/ (дата обращения: 24.06.2020)
17. About Java [Электронный ресурс]. URL: https://www.java.com/ru/about/ (дата обращения: 24.06.2020)
18. About Python [Электронный ресурс]. URL: https://www.python.org/about/ (дата обращения: 24.06.2020)
19. IEE SPECTRUM [Электронный ресурс]: Interactive: The Top Programming Languages — 2019. URL: https://spectrum.ieee.org/static/interactive-the-top-programming-languages-2019 (дата обращения: 24.06.2020)

Журнал «Современная наука: актуальные проблемы теории и практики»

Г. Владивосток