ПЛЕНАРНЫЕ ДОКЛАДЫ

УДК 004.424

<https://doi.org/10.36906/AP-2020/01>

ОТ ПАРАДИГМЫ ИМПЕРАТИВНОГО ПРОГРАММИРОВАНИЯ К ПАРАДИГМЕ КВАНТОВЫХ ВЫЧИСЛЕНИЙ И ДАЛЕЕ

Копыльцов А. А.

*Санкт-Петербургский государственный электротехнический
университет «ЛЭТИ» им. В.И. Ульянова (Ленина),
г. Санкт-Петербург, Россия*

Копыльцов А. В.

*д-р техн. наук,
Санкт-Петербургский государственный университет
аэрокосмического приборостроения,
г. Санкт-Петербург, Россия*

Аннотация. Рассматриваются различные парадигмы программирования (императивного, функционального, логического, объектно-ориентированного и др.). Особое внимание уделяется параллельному программированию (императивному, логическому, квантовому и др.), как наиболее перспективному направлению развития программирования.

Ключевые слова: программирование, парадигма, параллельное программирование, квантовое программирование.

Введение

Под парадигмой понимаем научные установки, представления и термины, на которых основываются современные научные исследования, в частности, в области программирования. Из истории известно, что парадигмы могут быть правильные или нет. Например, много веков существовала геоцентрическая парадигма устройства мира и парадигма плоской земли и небесного свода. Исторически в программировании сложилось несколько парадигм, каждая из которых предназначена для решения определенного класса задач на определенных вычислительных системах [3].

Парадигма императивного программирования

Первоначально возникла парадигма императивного программирования, которая сводится к оператору присваивания и оператору goto, т.е. процесс вычисления осуществляется по шагам и полностью контролируется на каждом шаге. Эта парадигма была создана для первых компьютеров 1940-1970 гг. и была реализована на таких известных языках программирования как Fortran (1954), Algol (1960), Pascal (1970), C (1972), ICON (1974) и др. Основная область использования — расчеты по заданным формулам или алгоритмам, как для военных, так и гражданских целей [3].

Парадигма функционального программирования

В основе этой парадигмы лежит понятие функции, т.е. идея представления программы как сложной функции. В 1960-1980 гг. большие надежды возлагались на функциональное программирование при разработке систем искусственного интеллекта, в основе которых лежали исследования функционирования мозга человека. К языкам функционального программирования можно отнести такие языки, как Lisp (1958), РЕФАЛ (1968), Scheme (1975), FP (1977), ML (1978), Miranda (1985), Haskell (1990) и др. Основная область использования — класс задач, трудно формулируемых в терминологии последовательных операций, в частности, задачи искусственного интеллекта, естественного языка, зрения и др. [1, 2, 5].

Парадигма логического программирования

В основе этой парадигмы лежит идея, что проблема описана в виде фактов (аксиом) и логических формул и ее решение может быть получено путем логического вывода, т.е. механизм логического доказательства применим ко всей программе, на основе аксиом и правил вывода (дизъюнктов). Первым языком логического программирования был Prolog (1971). В 1970-1990 гг. логическое программирование развивалось по трем направлениям. Первое направление связано с модификацией языка Prolog, путем внесением модульности, использованием более мощных логических средств и др. К языкам этого направления можно отнести Prolog II (1980), IC-Prolog (1982), MProlog (1983), T-Prolog (1983), LQF (1984), Goedel (1992) и др. Второе направление можно характеризовать, как комбинация логического и функционального программирования. Среди языков этого направления можно выделить LOGLISP (1982), LCA (1982), LEAF (1985), Mercury (1993) и др. Третье направление характеризуется использованием конструкций распараллеливания. Языками, относящимися к этому направлению являются Concurrent Prolog (1983), PARLOG (1983), GHC (1986) и др. Логическое программирование используется при решении задач искусственного интеллекта, задач обработки естественного языка, зрения и др. [1, 2, 5].

Парадигма объектно-ориентированного программирования

Эта парадигма предназначена для решения задач, в которых можно выделить объекты и установить связи (обмен информацией) между ними. К языкам объектно-ориентированного программирования обычно относят языки, позволяющие определить объекты, которые принадлежат классам и обладают свойствами инкапсуляции, полиморфизма, наследования. Можно выделить три группы языков: чистые, гибридные и урезанные языки объектно-ориентированного программирования. К первым можно отнести Simula (1962), Smalltalk (1972), Beta (1975), Self (1986), Cecil (1992) и др. Ко второй группе относятся C++ (1983), Object Pascal (1984) и др. К третьей — Java (1995), C# (2000) и др. Различие между тремя группами языков состоит в том, что в первой группе языков наиболее чисто отражена методология объектно-ориентированного программирования. Вторая группа языков появилась путем внедрения в широко известные императивные языки программирования конструкций объектно-ориентированного программирования. Третья группа получилась из второй путем удаления конструкций, которые оказались не нужны или представляли опасность с точки зрения объектно-ориентированного программирования. Основная область использования — моделирование взаимоотношений между объектами различных предметных областей.

Парадигма параллельного программирования

В связи с развитием вычислительной техники и, в особенности, с появлением многоядерных процессоров очень быстро стало развиваться параллельное программирование. Появились императивное параллельное программирование, логическое параллельное программирование и др.

Особенностью императивного параллельного программирования является то, что в программе можно явно выделить фрагменты (модули), которые могут быть выполнены параллельно. Если в программе такие модули можно выделить, то на многопроцессорной вычислительной системе время, затрачиваемое на прогон программы, может быть существенно сокращено. При проведении параллельных расчетов возникает проблема синхронизации взаимодействия между выполняемыми одновременно фрагментами программы, которая успешно решается в современных вычислительных системах. Обычно выделяют параллелизм на следующих уровнях:

- микрокоманд;
- операторов;
- циклов (итераций);
- подпрограмм (функций, процедур);
- потоков управления;
- процессов;
- приложений.

При написании программ, предназначенных для параллельного исполнения, используют явные конструкции. Можно выделить основные подходы параллельного программирования такие, как программирование на универсальном языке (например, на языке программирования Ada), программирование на широко используемом языке программирования (C, C++, Pascal и др.), в который добавлены конструкции, позволяющие проводить параллельные вычисления, автоматическое распараллеливание последовательных программ компиляторами, например. К языкам программирования, содержащим явные средства параллелизма, можно отнести Algol-68 (1968), Concurrent Pascal (1972), Modula-2 (1978), CSP (1978), Edison (1980), Occam (1982), Ada (1983), Linda (1985), Oblig (1993) и др. Область применения такого подхода — моделирование, системы реального времени, обработка юольших массивов данных и др.

Логическое параллельное программирование допускает естественное распараллеливание, поскольку процессы доказательства можно реализовывать одновременно. К языкам логического параллельного программирования можно в первую очередь отнести Concurrent Prolog (1983).

Парадигма квантовых вычислений

Под вантовым компьютером обычно понимают вычислительное устройство, которое при расчетах использует такие понятия квантовой механики, как квантовая суперпозиция, квантовая запутанность, квантовый параллелизм и др. Если в обычном компьютере есть биты (два устойчивых состояния обозначаемых либо 0, либо 1), то в квантовом — кубиты (квантовые биты, способные принимать и 0 и 1 одновременно). Это теоретически позволяет одновременно обрабатывать все возможные состояния, что для некоторых алгоритмов процесс получения результатов ускоряется в несколько раз [4].

Впервые идею квантового компьютера высказал Р. Фейнман на конференции в 1959 г. Позже подобные идеи высказывались во многих странах. Построение квантового компьютера как физического прибора — одна из важнейших задач современной физики.

Существуют различные подходы при построении компьютеров. В качестве логических кубитов используются:

- направления электронного (или ядерного) спина в данной квантовой точке на полупроводнике;
- наличие или отсутствие заряда (электрона) в определенной точке на полупроводнике;
- сверхпроводящие элементы (присутствие/отсутствие куперовской пары в определенной точке пространства);
- основное/возбужденное состояние внешнего электрона в ионе;
- квантовые состояния света и др.

Проблемы при построении таких компьютеров — точность измерения, минимизация внешних воздействий способных разрушить квантовую систему, минимизация числа ошибок и др.

Для квантовых компьютеров разрабатываются языки программирования, например, Quipper, в основе которого лежит язык Haskell

Область применения квантовых компьютеров — исследование многочастичных систем в области физики, химии, биологии, экономики, а также прогнозирование погоды, создание новых лекарств, финансовое моделирование и др.

В настоящее время разработаны квантовые компьютеры, состоящие из нескольких десятков кубитов. Цель — построить 100-кубитный квантовый компьютер, позволяющий решать задачи, которые не может решить классический компьютер.

Биокомпьютеры

Одна из важных проблем это построение компьютера аналогичного мозгу человека. С точки зрения анатомии устройство мозга человека изучено достаточно подробно. Известно строение и расположение нейронов, выделены различные области коры головного мозга человека, установлены связи между ними и пр. Исследования проводились на различных живых объектах (дождевые черви, лабораторные мыши, кролики, кошки, собаки, приматы и пр.). Однако остается загадкой, как мозг работает? Для решения этой проблемы используются различные подходы. В частности, были разработаны искусственные нейронные сети. Была выдвинута идея, что для проявления интеллекта нужно большое количество элементов, сопоставимое с количеством нейронов в мозге человека (около 14 миллиардов). Проводились эксперименты по созданию компьютера в пробирке (растворе) на основе ЯМР. Проводимые эксперименты в этой области пока не превосходят результатов, получаемых на классических компьютерах.

Заключение

Рассмотрены различные парадигмы программирования: парадигма императивного программирования, парадигма функционального программирования, парадигма логического программирования, парадигма объектно-ориентированного программирования, парадигма параллельного программирования, парадигма квантовых вычислений, парадигма биокомпьютера и др. Видно, что каждая из парадигм предназначена для решения определенного круга задач, возникающих на различных этапах развития вычислительной техники. В последние десятилетия в связи с развитием физики и информационных технологий получили развитие области, связанные с квантовыми вычислениями и биокомпьютерами.

Литература

1. Копыльцов А. А., Копыльцов А. В. Алгоритм обработки слабо формализованной информации, поступающей от технических систем // Известия СПбГЭТУ «ЛЭТИ». Серия «Информатика, управление и компьютерные технологии. 2012. №8. С. 30-36.
2. Костров Б. В., Ручкин В. Н., Фулин В. А. Искусственный интеллект и робототехника. М: ДИАЛОГ–МИФИ. 2008.
3. Одинцов И. О. Профессиональное программирование. Системный подход. СПб: БХВ-Петербург. 2004.
4. Прилипко В. К. Физические основы квантовых вычислений: динамика кубита. СПб: Лань. 2019.
5. Рутковский Л. Методы и технологии искусственного интеллекта. М: Горячая линия–Телеком. 2010.

©Копыльцов А. А., Копыльцов А. В., 2020