ИСПОЛЬЗОВАНИЕ ЭВОЛЮЦИОННЫХ МЕТОДОВ ДЛЯ ФОРМИРОВАНИЯ МУЛЬТИВЕРСИОННОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

О. А. Барышева, Л. В. Ермолаева

Институт торговли и сферы услуг
Сибирского федерального университета, Красноярск, Россия
E-mail: s_ol_a@mail.ru, elv1112@mail.ru

В данной статье приведен обзор семейств алгоритмов для формирования мультиверсионного программного обеспечения. Особенность этих семейств заключается в том, что механизмы, лежащие в основе их деятельности, были получены в результате исследования процессов, протекающих в «живом» мире.

THE USE OF EVOLUTIONARY METHODS OF MULTI-VERSION SOFTWARE

O. A. Barysheva, L. V. Ermolaeva

This article provides an overview of the families of algorithms for the formation of multi-version software. The peculiarity of these families is that the mechanisms underlying their activities were obtained as a result of studying the processes occurring in the "living" world.

Идея мультиверсионного программирования, как способ воплощения программной отказоустойчивости, была введена А. Авижиенисом в 1977 году [1-3].

Под независимой генерацией программ понимается формирование конструирования программных средств (ПС) таким образом, чтобы каждая из N, участвующих в разработке отдельных программных модулей, групп не взаимодействовала с другой по отношению к процессу реализации ПС. В связи с этим каждая из групп разработчиков должна по возможности использовать различные алгоритмические, инструментальные и языковые средства проектирования.

При создании программного обеспечения используются различные модели и методы принятия решений. Это позволяет из множества решений выбрать программное обеспечение, отвечающее заданным требованиям проектировщика.

В итоге отказоустойчивое программное обеспечение создается по следующему принципу: отвергаются результаты версии, не совпадающие с половиной версий и еще одной или не полученные вовремя, т.е. с применением системы согласования [4].

Создание мультиверсионного программного обеспечения строится по двум основным принципам:

1. Программное обеспечение состоит из модулей.

2. В каждом модуле есть несколько независимых версий, которые можно комбинировать.

В свою очередь, оптимальный состав мультиверсионного программного обеспечения должен создать возможность изменения в реальном времени структуры программного обеспечения, подстраиваясь к изменяющимся условиям работы. В связи с этим были созданы алгоритмы, которые способны решать задачи подобной сложности за ограниченное количество времени.

Для принятия решений в выборе программного обеспечения, отвечающего требованиям проектировщика, используются различные модели и методы. Существует несколько семейств алгоритмов, которые объединяет то, что механизмы, лежищие в основе их деятельности, были получены в результате исследования процессов, протекающих в природе, нас окружающей. К ним относятся: генетический алгоритм, метод роя частиц, муравьиный алгоритм и др.

Вначале рассмотрим генетический алгоритм (genetic algorithm). Это эвристический алгоритм поиска, который для решения задачи использует механизмы, аналогичные естественному отбору в природе. Генетические алгоритмы являются универсальным вычислительным средством для решения серьезных математических задач

При построении данного алгоритма используются следующие понятия: ген, хромосома, популяция, приспособленность, селекция, скрещивание, мутация, особи, поколение.

Этапы построения генетического алгоритма заключаются в следующем:

– Создание начальной популяции. Случайным образом создается «начальная популяция» – n начальных векторов. Тогда решение задачи может быть представлено в виде вектора («хромосома»).

– Вычисление функций приспособленности для особей популяции (оценивание). Начальные векторы оцениваются с использованием «функции приспособленности», в результате чего каждому вектору присваивается определённое значение, которое определяет вероятность выживания организма, представленного данным вектором.

– Выбор индивидов из текущей популяции («селекция»). По результатам оценивания выбираются вектора («селекция») для «скрещивания».

– «Скрещивание» и\или «мутация». Применяя «генетические операторы» «скрещивания» и «мутации», создается следующее «поколение».

– Вычисление функций приспособленности для всех особей. Векторы – особи следующего поколения – также проходят этапы оценивания, селекции, применения генетических операторов и т.д.

– Формирование нового «поколения». В результате нескольких жизненных циклов («поколений») происходит формирование нового «поколения».

– Конец цикла, при условии выполнения поставленных критериев.

– Начало цикла, в противном случае.

Поскольку данный алгоритм решения задачи основывается на практическом методе, он не является гарантированно точным или оптимальным. Однако он дает достаточно хорошее решение поставленной задачи в большинстве случаев.

Для формирования мультиверсионного программного обеспечения применяется также метод роя частиц, который был разработан Джеймсом Кеннеди и Расселом Эбер-хартом в 1995 году. Данный алгоритм является методом численной оптимизации, при использовании которого не требуется знать точного градиента оптимизируемой функции. Возможные решения, называемые частицами или агентами, перемещаются в пространстве к наилучшему найденному в этом пространстве решению, всё время находящемуся в изменении из-за нахождения агентами более выгодных решений. Изначально этот алгоритм применялся для исследований социального психолога Кеннеди, но самое большое распространение этот алгоритм смог получить при решениях задач оптимизации различных нелинейно-многомерных уравнений.

Частицы перемещаются в пространстве решений согласно формуле, в результате чего находят наиболее выгодное положение, по отношению к соседям.

Состояние частицы в настоящий момент времени характеризуется координатами в пространстве решений и вектором скорости перемещения, которые выбираются случайным образом на этапе подготовке к работе. Каждая частица-агент хранит координаты лучшего из найденных решений.

В соответствии со сведениями о найденных оптимальных решениях изменяются направление и длина вектора скорости каждой из частиц на каждом этапе.

$$v_n^{i+1} = v_n^i + c_1 \cdot R \cdot (\rho_n - x_n) + c_2 \cdot R \cdot (g_n - x_n),$$

где υ – вектор скорости частицы (v^n – его n-ый элемент),

c_1, c_2 – постоянные ускорения,

ρ_n – наилучшая точка, найденная частицей,

g_n – наилучшая точка для всех частиц системы,

x – текущее положение частицы, а функция R возвращает случайное число от 0 до 1 включительно, по информации в статье [5].

Вычисляется направление вектора ν, и частица перемещается в точку $x = x + \upsilon$. Значения лучших точек для каждой частицы и для всех частиц в целом обновляются и цикл повторяется. При условии выполнения поставленных критериев цикл завершается.

Т.к. метод роя частиц имеет лишь один оператор — вычисление скорости, это делает его более быстрым, а также в методе роя можно легко определить достижение точки глобального минимума.

Муравьиный алгоритм - это эффективный методов для решения поиска путей на графах. Алгоритм заключается в изучении и применении поведения муравьёв, которые ищут путь от муравейника к источнику апраитания и представляет собой метаэвристическую оптимизацию. Первая версия алгоритма, предложенная доктором наук Марко Дориго в 1992 году.

Муравьи специалисты в области передачи информации. Вся их жизнь построена на этом. Без умения передавать информацию невозможна была бы такая скоординированность их действий. Главный язык, с помощью которого общаются муравьи, химический. Муравьи, выделяя пахучие вещества, или, как их называют, феромоны, якобы обозначают ими направление пути, объявляют

тревогу и подают другие сигналы. Рассмотрим такой пример. Вот движется колонна рабочих муравьев по тропе. Стоит только появиться препятствию, мешающему доставлять листья в муравейник, и хрупкое равновесие будет нарушено. Листья должны поступать в муравейник регулярно, необходимо строжайше придерживаться заведенного ритма. Но оказавшись перед препятствием, муравьи распространяют пахучие вибрации, вызывая возбуждение идущих следом. Запах сигнала вызывает у них непременное желание оттащить это препятствие и все ближайшие муравьи группируются. Возбуждение нарастает, новые особи добавляют пахучих веществ в воздух. Так рабочие стимулируют друг друга. Но как только препятствие будет устранено и путь восстановлен, муравьи продолжат движение. Такое поведение можно описать с помощью моделей, в которой используется механизм стигмерции. Стигмергия является одной из форм самоорганизации, создавая сложные, казалось бы, интеллектуальные структуры, но без какого-либо планирования, контроля, или даже прямой связи между индивидами. И именно этот механизм положен в основу алгоритмов муравьиной колонии [6].

Алгоритм муравьиной колонии можно разбить на три этапа: принятие решения муравьями, обновления значений феромона, дополнительные действия.

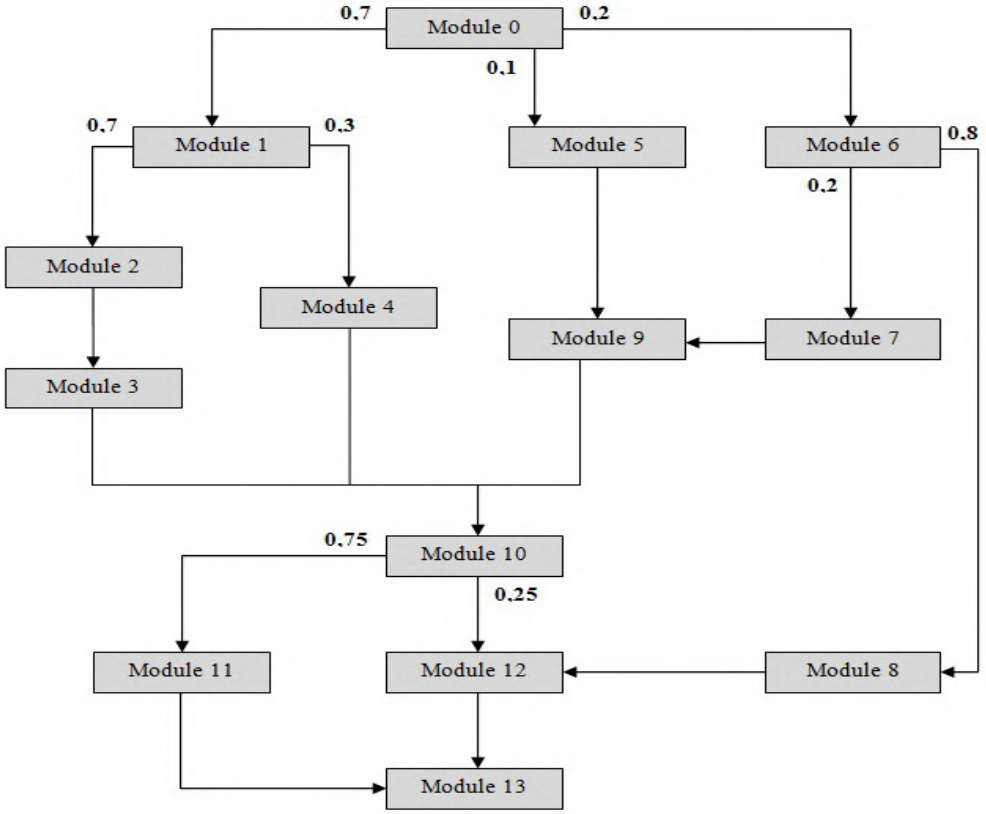
На первом этапе происходит управление агентами, которые одновременно и независимо друг от друга осуществляют движение по узлам графа, по заданным условиям задачи. Агенты принимают решение на основе вероятностного правила, которое применяется на основании информации о значении феромона и эмпирической привлекательности возможностей. Так постепенно выбирается решение для поставленной задачи. При этом происходит оценка принятого решения и выявляется, какое количество феромона необходимо отложить.

Второй этап это процесс, на котором вносятся изменения в значение феромона. Увеличение показателя осуществляется в результате откладывания феромона агентами во время поиска пути, уменьшение происходит в результате применения операции испарения феромона. Увеличение количества феромона позволяет повысить вероятность того что в дальнейшем, при выборе пути, будет выбран тот маршрут, который является наиболее эффективным из имеющихся альтернатив. Процедура испарения феромона введена, как механизм, позволяющий избежать быстрой сходимости к одному решению, что позволяет избежать скатывания в область локального оптимума благодаря способности забывать о плохих решениях, принятых в результате работы алгоритма.

Третий этап это локальный поиск, сбор и анализ информации, исходя из которой необходимо определить необходимо ли прилагать дополнительные усиления наиболее перспективных областей поиска, или же надо увеличить область поиска.

Томас Штютцле и Хольгер Хоос предложили муравьиный алгоритм Maxmin Ant System, в котором добавляются граничные условия на количество феромонов (τ_{min},τ_{max}). Феромоны откладываются только на глобально лучших или лучших в итерации путях. Все рёбра инициализируются значением τ_{max} [7].

Изменение алгоритма будет сделано на основе алгоритма MAX-MIN Ant System [8;9], так как этот алгоритм показывает хорошие результаты на любых классах задач и является самым изученным [10;11].



Структура модулей

На рисунке изображены модули, соединенные между собой графами, с вероятностями перехода от одного к другому модулю. В каждом модуле есть некоторое количество версий со значением стоимости и надежности. В зависимости от выбранных версий в модуле рассчитывается его стоимость и надежность, а с помощью графа рассчитывается стоимость и надежность всей ПС.

На этапе реализации минимального решения отличий от алгоритма MAX-MIN Ant System нет. Выбор версии с минимальными решением по классическому правилу выбора, знакомое нам еще по алгоритму Ant System [9].

Итак, модифицируемый алгоритм муравьиной колонии с новым правилом для решения задачи формирования мультиверсионного программного обеспечения с учетом ее специфики, показал хорошие результаты работы. На основе разработанного алгоритма был создан программный комплекс для реализации мультиверсионного программного обеспечения, с помощью которого провели эксперименты и собрали статистику для анализа работы начального и измененного алгоритма муравьиной колонии. Исследована задача практического применения алгоритма муравьиной колонии для создания мультиверсионного программного обеспечения и проведен анализ результатов. В итоге были получены данные, которые показывают превосходство модифицированного алгоритма.

СПИСОК ЛИТЕРАТУРЫ

1. *Avizienis A.* The N-Version approach to fault –tolerant software // IEEE Trans. On Software Engineering. 1985. Vol. SE11. № 12. P. 1491–1501.

2. *Lyu M. R*. Handbook of Software Reliability Engineering / IEE Computer Society Press and McGraw –Hill Book Company, 1996. 819 p.

3. *Lyu M. R.* Software Fault Tolerance. John Wiley&Sons Ltd, 1996.

4. *Ковалев И. В., Ступина А. А., Царев Р. Ю., Волков В. А.* Применение COM-технологии для реализации мультиверсионного программного обеспечения систем управления и обработки информации // Приборы и системы. Управление, контроль, диагностика. 2007. № 3. С. 18–22.

5. *Баранюк В.В., Смирнова О.С.* Детализация онтологической модели по роевым алгоритмам, основанным на поведении насекомых и животных // International Journal of Open Information Technologies. 2015. № 12. С.18-27.

6. *Зайцев А.А, Курейчик В. В., Полупанов А. А.* Обзор эволюционных методов оптимизации на основе роевого интеллекта // Известия ЮФУ. Технические науки. 2010. № 12. С. 7-12.

7. *Stützle T., Hoos H.,* “MAX-MIN Ant System and local search for the traveling salesman problem” // IEEE International Conference on Evolutionary Computation. 1997. P. 309-314.

8. *Dorigo M., Stutzle Th.* Ant Colony Optimization // Massachusetts Institute of Technology. 2004.

9. *Ковалев И. В. [и др.]* Использование метода роя частиц для формирования состава мультиверсионного программного обеспечения // Приборы и системы. Управление, контроль, диагностика. 2013. № 3. С. 1–6.

10. *Corne D., Dorigo M., Glover F.* New Ideas in Optimization / McGrav – Hill, 1999. 314 p.

11. *Штовба С. Д.* Муравьиные алгоритмы // Математика в приложениях. 2003. № 4 (4). С. 70–75.