УДК 658.512.001.56

МОДИФИКАЦИЯ СТАНДАРТНОГО АЛГОРИТМА МУРАВЬИНОЙ КОЛОНИИ ДЛЯ РЕШЕНИЯ ЗАДАЧИ ФОРМИРОВАНИЯ МУЛЬТИВЕРСИОННОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Д. И. Ковалев, А. В. Клименко, Е. В. Соловьев, Е. В. Туева

Сибирский государственный аэрокосмический университет имени академика М. Ф. Решетнева
Российская Федерация, 660014, г. Красноярск, просп. им. газ. «Красноярский рабочий», 31
E-mail: kovalev.fsu@mail.ru

Предложен алгоритм муравьиной колонии с новым решающим правилом для решения задачи формирования мультиверсионного программного обеспечения. Учтена специфика постановки задачи как задачи покрытия множества. Разработан программный комплекс на основе предложенных алгоритмов для формирования состава мультиверсионного программного обеспечения. Проведены эксперименты и собрана статистика для сравнения работы стандартного и модифицированного алгоритма муравьиной колонии. Получены данные о том, что модифицированный алгоритм демонстрирует лучшие значения целевой функции, а также большую скорость нахождения решения.

Ключевые слова: мультиверсионное программное обеспечение, алгоритм муравьиной колонии, программный комплекс, задача покрытия множества.

MODIFICATIONS TO THE STANDARD ANT COLONY ALGORITHM FOR SOLVING THE PROBLEM OF FORMATION MULTIVERSIONED VIEWS SOFTWARE

D. I. Kovalev, A. V. Klimenko, E. V. Soloviev, E. V.Tueva

Siberian State Aerospace University named after academician M. F. Reshetnev
31, Krasnoyarsky Rabochy Av., Krasnoyarsk, 660014, Russian Federation
E-mail: kovalev.fsu@mail.ru

The ant colony algorithm with a new decision rule for solving the problem of multisession software formation is offered. The specificity of the task as the task cover set is taken into account. The software complex on the basis of the proposed algorithms for the formation of the composition of multisession software is developed. The experiments have been carried out and statistics to compare performance standard and modified ant colony algorithm have been collected. The data that the modified algorithm demonstrates the best values of the objective function, and more speed to find solutions have been obtained.

Keywords: multisession views software, ant colony algorithm, software system, the task of covering many.

В настоящее время существуют два взаимно дополняющих друг друга подхода для разработки устойчивого к сбоям программного обеспечения (ПО) [1–3]: безопасное программирование и отказоустойчивые системные архитектуры. Нас интересует подход с использованием отказоустойчивых архитектур. В его основе лежит резервирование аппаратных и программных средств и выявление ошибочных результатов путем сравнения контрольных данных.

В большинстве случаев технология разработки устойчивого к сбоям оборудования базируется на методе модульного резервирования, когда модуль оборудования дублируется несколько раз. Выходные данные каждого модуля сравниваются между собой [4; 5]: если один из модулей выходит из строя или его выходные данные отличаются от выходных данных других модулей, эти данные игнорируются. Если сразу восстановить сбойный модуль невозможно, система автоматически отключает его и продолжает свою работу с оставшимися модулями.

Существует две методологии для создания отказоустойчивого программного обеспечения: блоки восстановления и мультиверсионное программирование (N-version programming или multiversion programming) [5–7]. При использовании методологии мультиверсионного программирования осуществляется разработка мультиверсионного программного обеспечения (NVC – n-version software), которое строится на основе множества версий с общей спецификацией, созданных разными командами разработчиков. Эти версии выполняются параллельно. Результаты их работы сравниваются с помощью системы согласования [8], результаты версии, не совпадающие с половиной версий и еще одной или не полученные вовремя, отвергаются. Это наиболее часто используемый подход к созданию отказоустойчивого программного обеспечения. Именно этот подход является одним из предметов исследования данной работы.

Эффективность технологий проектирования программ непосредственно влияет на итоговую совокупность затрат на создание программных комплексов [9]. Кроме того, с развитием и повсеместным внедрением автоматизированных комплексов, которые работают с объектами в реальном времени, очень важное значение получают алгоритмы, которые позволяют в реальном времени и с заданными ограничениями эффективно искать решения оптимизационных задач. Одной из таких задач является задача формирования оптимального состава мультиверсионного программного обеспечения, благодаря решению которой можно в реальном времени изменять структуру программного обеспечения, подстраиваясь к изменяющимся условиям работы [10]. Проблемой при решении таких задач является их сложность, поэтому были созданы алгоритмы, которые способны решать задачи подобной сложности за ограниченное количество времени [11].

Существует несколько семейств алгоритмов, которые объединяет то, что механизмы, которые лежат в основе их деятельности, были получены в результате исследования процессов, протекающих в «живом» мире [12]. Эти алгоритмы получили название биоинспиративных алгоритмов. Одним из таких алгоритмов является алгоритм муравьиной колонии [13; 14]. В данной работе исследуется его применение для задачи формирования состава мультиверсионного программного обеспечения.

Предложенная модификация выполнена на основе алгоритма MAX-MIN Ant System [12; 15], так как помимо демонстрации хороших результатов практически на любых классах задач данный алгоритм является одним из самых изученных алгоритмов муравьиной колонии [16; 17]. Для решения задачи формирования состава мультиверсионного программного обеспечения алгоритм MAX-MIN Ant System применялся и без какой-либо модификации [18], но в этом случае при расчете не будет использоваться весь объем информации, который несет постановка задачи, в частности, для мультиверсионного ПО с динамической архитектурой среды исполнения [19]. Предлагается внести ряд изменений, которые позволят алгоритму улучшить найденные решения за счет более полного учета знаний (сведений) о поставленной задаче.

Итак, структура ПО задается набором модулей, соединенных определенным образом, образующих граф переходов с вероятностями перехода из одного модуля в другой. В каждом модуле существует определенное количество версий со значением надежности и стоимости. На основании выбранных версий в модуле вычисляется его надежность и стоимость, а благодаря наличию графа программы мы имеем возможность вычислить надежность и стоимость всей программной структуры. Также в условиях задачи должны присутствовать ограничения, накладываемые на надежность и стоимость итогового решения.

Упрощенно ACO-алгоритмы можно представить как повторение трех процедур [12]: построение решения, обновление значений феромона, дополнительные действия:

```
procedure ACOMetaheuristic
    ScheduleActivities
    ConstructAntsSolutions
    UpdatePheromones
    DaemonActions /*опционально*/
    end-ScheduleActivities
end-procedure
```

На этапе построения решения происходит управление муравьями (агентами), которые одновременно и независимо друг от друга осуществляют движение и построение решений.

Обновление феромона представляет собой процесс, в результате которого вносятся изменения в показатели феромона, увеличение показателя осуществляется в результате увеличения показателя феромона агентами. Также существует процедура испарения феромона. Это механизм, позволяющий избежать быстрой сходимости к одному решению. Стоит отметить, что для данной задачи все значения феромона мы ассоциируем непосредственно с характеристиками версий модулей.

Дополнительные процедуры представляют собой реализацию действий, которые невозможно выполнить одному агенту. Примером таких действий может быть операция локального поиска или сбор и анализ общей информации.

В работе приведены основные особенности и формулы, используемые при реализации алгоритма MAX-MIN Ant System (MMAS) для минимизируемой функции, которая зависит от одной переменной.

Как сказано выше, алгоритм представляет три процедуры: построение муравьями решения, обновление значений феромона, дополнительные действия. Для данной задачи все значения феромона мы ассоциируем непосредственно с версиями модулей ПО. Но так как по условиям задачи каждый модуль должен быть представлен как минимум один раз, мы делим процедуру построения решения на две части: построение минимального решения и добавление дополнительных элементов. Поэтому в упрощенном виде наш алгоритм будет выглядеть следующим образом:

```
procedure ACOMetaheuristic
ScheduleActivities
    ConstructAntsMinSolutions
    UpgradeAntsSolutions
    UpdatePheromones
    DaemonActions % optional
    end-ScheduleActivities
end-procedure
```

На этапе построения минимального решения никаких отличий от стандартного алгоритма MAX-MIN Ant System не будет. Решающее правило, по которому осуществляется выбор версии модуля при построении минимального решения, представляет собой классическое правило выбора, знакомое нам еще по алгоритму Ant System [15].

Что касается процесса построение решения, то рекомендуется реализовать его последовательно, так как количество версий, образующих решение, может быть различным при поиске для каждого агента. Последовательное построение решения позволит избежать постоянных проверок на завершение поиска.

После того, как будут построены минимальные решения, начинается этап улучшения решения. Для каждого агента производится попытка добавить избыточный модуль в итоговое решения для того, чтобы улучшить показатель его надежности, при этом нужно помнить о существующих ограничениях, чтобы получаемые решения были приемлемыми. В решающее правило вносятся изменения для того, чтобы в нем содержалась информация о вероятности использования модуля и конкретной версии в модуле. Это необходимо для того, чтобы добавлять версии именно в те модули, которые имеют критическое значение для всей программной системы. Также стоит отметить, что на этапе улучшения решения выбор осуществляется среди всех версий всех модулей, которые не задействованы в программе. Эти изменения, вносимые в решающее правило алгоритма, позволяют задействовать нам всю информацию из условий постановки задачи формирования мультиверсионного ПО с динамической архитектурой.

Предложенная в данной работе модификация муравьиного алгоритма для задачи формирования состава отказоустойчивого программного обеспечения была реализована в программном комплексе, получившем название ACO N-version software creator. Данное программное приложение написано на языке программирования C# с использованием среды программирования приложений Microsoft Visual Studio 2012. Это позволило использовать возможности, предоставляемые объектно-ориентированным подходом при разработке программного обеспечения, а также задействовать Windows Forms Framework при разработке графического интерфейса пользователя, знакомого любому пользователю персонального компьютера, работающего под управлением операционной системы Windows.

Пользователем задаются все параметры задачи. Определяются модули будущей мультиверсионной программы, их название и состав. Каждый модуль содержит некоторое количество версий, каждая из которых имеет название, а также показатель надежности и стоимости. Пользователь формирует структуру связей между будущими модулями программы, а также вероятности перехода от одного модуля к другому. Задаются ограничения, которые накладываются на программу, в виде пороговых значений надежности и стоимости. Задается цель поиска. Это может быть минимизация стоимости программы с учетом ее надежности или максимизация показателя надежности с учетом заданного показателя максимальной стоимости мультиверсионного ПО.

Разработанный программный комплекс позволяет проводить эксперименты на различных задачах формирования оптимального состава мультиверсионного программного обеспечения. Однако при программной реализации муравьиного алгоритма были внесены изменения, которые позволили организовать контроль исполнения алгоритма и учесть особенность задачи формирования мультиверсионного программного обеспечения. Особенности реализации муравьиного алгоритма в этом случае удобно рассмотреть при анализе функции CalculateSolution, которая выглядит следующим образом:

```
public ProgSolution CalculateSolution()
{
   AntManager.ConstructMinimumSolution();
   AntManager.ConstructSolution();
   var isBestUpdated = AntManager.UpdateBestSolution();
   PheromoneManager.UpgradeValues(AntManager.BestAnt);
   AntManager.ResetAnts();
   PheromoneManager.ResetPheromones(isBestUpdated);
   if (AntManager.BestAnt != null)
      return AntManager.BestAnt.Solution.Clone();
   return null.
```

1. AntManager.ConstructMinimumSolution () – задача формирования состава мультиверсионного программного обеспечения имеет ограничение, которое накладывается на найденное решение, что задается отдельной функцей, которая гарантирует соблюдение данного ограничения. Это позволит рассчитать начальные параметры решения.

2. AntManager.ConstructSolution() – функция осуществляет процесс улучшения найденных решений посредством добавления избыточных версий; особенностью реализации является последовательная работа с агентами, что позволяет избежать лишних проверок.

3. AntManager.UpdateBestSolution() – функция проводит анализ найденных решений. Если было найдено решение, качество которого лучше ранее найденного решения, происходит его сохранение в переменную и функция возвращает логическую 1, в противном случае возвращается 0.

4. PheromoneManager.UpgradeValues(AntManager.BestAnt) – функция осуществляет процесс обновления феромона: испарение существующего феромона и увеличение феромона на лучшем найденном решении благодаря переданному значению.

5. AntManager.ResetAnts() – для всех агентов выполняется операция очистки найденных решений, что позволяет использовать их на следующей итерации поиска без дополнительного выделения памяти.

6. PheromoneManager.ResetPheromones(isBestUpdated) – функция проверяет количество итераций без улучшения, в случае превышения порогового значения происходит сброс всех показателей феромона в их максимальное значение, что позволяет направить операцию поиска в еще не исследованные области.

7. Если было найдено решение, то происходит его возврат для использования, в противном случае возвращается null.

Для проверки возможности применения алгоритма на реальном объекте была решена задача формирования мультиверсионного ПО системы обработки данных реального времени долговременной орбитальной

станции [20]. Модифицированный алгоритм продемонстрировал лучшие значения целевой функции, а также большую скорость при нахождении решения.

Таким образом, предложенный алгоритм муравьиной колонии с новым решающим правилом для решения задачи формирования мультиверсионного программного обеспечения с учетом ее специфики как задачи покрытия множества продемонстрировал лучшие результаты работы на тестовых задачах. Разработан программный комплекс на основе разработанного алгоритма для формирования состава мультиверсионного программного обеспечения, который позволил провести эксперименты и собрать статистику для сравнения работы стандартного и модифицированного алгоритма муравьиной колонии. На разработанных тестовых задачах проведены эксперименты и исследована эффективность модифицированного алгоритма в сравнении со стандартным, в результате чего были получены данные о том, что модифицированный алгоритм демонстрирует лучшие значения целевой функции, а также большую скорость нахождения решения. Исследована задача практического применения алгоритма муравьиной колонии для формирования состава мультиверсионного программного обеспечения и проведено сравнение результатов. В качестве реального объекта было рассмотрено программное обеспечение долговременных орбитальных станций. В результате сравнения стандартного и модифицированных алгоритмов были получены данные, которые показывают превосходство модифицированного алгоритма.

Библиографический список

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., Завьялова О. И., Лайков А. Н. Формирование избыточного программного обеспечения отказоустойчивых систем управления // Изв. вузов. Приборостроение. 2008. Т. 51, № 10. С. 30–34.

5. Ковалев И. В., Юнусов Р. В. Мультиверсионный метод повышения программной надежности информационно-телекоммуникационных технологий в корпоративных структурах // Дистанционное и виртуальное обучение. 2003. № 2. С. 50–55.

6. Царев Р. Ю. Мультиверсионный подход к повышению отказоустойчивости программного обеспечения систем управления и обработки информации // В мире научных открытий. 2010. № 4 (10). Ч. 10. С. 82–84.

7. Ковалев И. В., Слободин М. Ю., Ступина А. А. Математическая постановка задачи проектирования N-версионных программных систем // Проблемы машиностроения и автоматизации. 2005. № 3. С. 16–23.

8. 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.

9. Ковалев И. В., Новой А. В., Штенцель А. В. Оценка надежности мультиверсионной программной архитектуры систем управления и обработки информации // Вестник СибГАУ. 2008. Вып. № 3 (20). С. 50–52.

10. Ковалев И. В., Новой А. В. Расчет надежности отказоустойчивых архитектур программного обеспечения // Вестник СибГАУ. 2007. Вып. 4 (17). С. 14–17.

11. Kovalev I. V., Dgioeva N. N., Slobodin M. Ju. The mathematical system model for the problem of multiversion software design // Proceedings of Modelling and Simulation, MS'2004. AMSE International Conference on Modelling and Simulation. Lyon-Villeurbanne, 2004.

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

13. Dorigo M., Maniezzo V., Colorni A. The Ant System: An Autocatalytic Optimizing Process. Technical Report No. 91-016 Revised, Politecnico di Milano, 1991. 103 p.

14. Colorni A., Dorigo M., Maniezzo V. An Inverstigation of Some Properties of an Ant Algorithm // Proceedings of the Parallel Problem Solving from Nature Conference (PPSN 92) (Brussels, Beldium) / R. Manner and B. Manderick (Eds.). Elsevier Publishing, 1992. P. 509–520.

15. 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.

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

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

18. Ковалев И. В., Карасева М. В., Соловьев Е. В. Модификация муравьиного алгоритма для задачи формирования мультиверсионного программного обеспечения // Вестник СибГАУ. 2014. Вып. № 1(53). С. 19–24.

19. Ковалев И. В., Царев Р. Ю., Прокопенко А. В., Соловьев Е. В. К вопросу реализации муравьиного алгоритма при выборе состава мультиверсионного программного обеспечения информационно-управляющих систем // Приборы и системы. Управление, контроль, диагностика. 2012. № 2. С. 1–4.

20. Кульба В. В., Микрин Е. А., Павлов Б. В. Проектирование информационно-управляющих систем долговременных орбитальных станций. М. : Наука, 2002. 343 с.

References

1. Avizienis A. The N-Version approach to fault – tolerant software. *IEEE Trans. On Software Engineering*. Vol. SE11, no. 12, December, 1985, 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. Kovalev I. V., Zav'jalova O. I., Lajkov A. N. [Formation of excessive soft-ware fault-tolerant control systems]. *Izvestija vysshikh uchebnykh zavedenij*. *Priborostroenie.* 2008, Vol. 51, no. 10, p. 30–34. (In Russ.)

5. Kovalev I. V., Junusov R. V. [Multiversioning-tion method for increasing software reliability information-communication technologies in corporate structures]. *Distantsionnoe i virtual'noe obuchenie.* 2003, no. 2, p. 50–55. (In Russ.)

6. Carev R. Ju. [Multiversioning approach to an increase resiliency of software program management systems and information processing]. *V mire otkrytij.* 2010, vol 10, no. 4, Ch. 10, p. 82–84. (In Russ.)

7. Kovalev I. V., Slobodin M. Ju., Stupina A. A. [Mathematical problem of designing N-versioned software systems]. *Problemy mashinostroenija i avtomatizatsii*. 2005, no. 3, p. 16–23. (In Russ.)

8. Kovalev I. V., Stupina A. A., Carev R. Ju., Volkov V. A. [Application of COM technology for realizations multiversioning soft-ware systems, control and information processing]. *Pribory i sistemy. Upravlenie, kontrol', diagnostika.* 2007, no. 3, p. 18–22. (In Russ.)

9. Kovalev I. V., Novoj A. V., Shtancel' A. V. [Evaluation of reliability multiversioning software architecture of management systems and information processing]. *Vestnik SibGAU.* 2008, vol. 20, no. 3, p. 50–52. (In Russ.)

10. Kovalev I. V., Novoj A. V. [Calculation of reliability of fault-tolerant software architectures]. *Vestnik SibGAU.* 2007, vol. 17, no. 4, p. 14–17. (In Russ.)

11. Kovalev I. V., Dgioeva N. N., Slobodin M. Ju. The mathematical system model for the problem of multiversion software design. *Proceedings of Modelling and Simulation, MS'2004* AMSE International Conference on Modelling and Simulation, MS'2004. Lyon-Villeurbanne, 2004.

12. Marco Dorigo, Thomas Stutzle. Ant Colony Optimization. *Massachusetts Institute of Technology,* 2004.

13. Dorigo M., Maniezzo V., Colorni A. The Ant System: An Autocatalytic Optimizing Process. Technical Report no. 91-016 Revised, Politecnico di Milano, Italy, 1991, 103 p.

14. Colorni A., M. Dorigo, V. Maniezzo An Inverstigation of Some Properties of an Ant Algorithm. *Proceedings of the Parallel Problem Solving from Nature Conference* (PPSN 92). Brussels, Beldium, R. Manner and B. Manderick (Eds.), Elsevier Publishing. 1992, p. 509–520.

15. Kovalev I. V., Solov'ev E. V., Kovalev D. I., Bahmareva K. K., Demsish A. V. [The use of particle swarm to form the composition of the multiverse-onnogo software]. *Pribory i sistemy. Upravlenie, kontrol', diagnostika,* 2013, no. 3, p. 1–6. (In Russ.)

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

17. Shtovba S. D. [Ant algorithms]. *Matematika v prilozhenijah*. 2003, no. 4 (4), p. 70–75. (In Russ.)

18. Kovalev I. V., Karaseva M. V., Solov'ev E. V. [Modification of the ant algorithm to the problem of forming multiversioning software]. *Vestnik SibGAU.* 2014, vol. 53, no. 1, p. 19–24. (In Russ.)

19. Kovalev I. V., Carev R. Ju., Prokopenko A. V., Solov'ev E. V. [On the Implementation of the ant algorithm for choosing the composition multiversioning software information management systems]. *Pribory i sistemy. Upravlenie, kontrol', diagnostika.* 2012, no. 2, p. 1–4. (In Russ.)

20. Kul'ba V. V., Mikrin E. A., Pavlov B. V. *Prektirovanie informatsionno-upravljajushhikh sistem dolgovremennykh orbital'nyh stantsij* [Design of information management systems of long-term orbital stations]. Moscow, Nauka Publ., 2002, 343 p.

УДК 004.056

МЕТОДИКА АНАЛИЗА И ОЦЕНКИ РИСКОВ ОРГАНИЗАЦИИ

И. З. Краснов[1], О. И. Карелин[2]

[1]Сибирский федеральный университет
Российская Федерация, 660041, г. Красноярск, просп. Свободный, 79
[2]Сибирский государственный аэрокосмический университет имени академика М. Ф. Решетнева
Российская Федерация, 660014, г. Красноярск, просп. им. газ. «Красноярский рабочий», 31
E-mail: bk_24@bk.ru, karelin@sibsau.ru

Стремительное развитие компьютерной сферы и высоких технологий в последние два десятилетия привело к тому, что информация приобрела конкретные финансовые, репутационные, временные и экономические выражения. В связи с этим для большинства организаций защита информации становится одной из приоритетных задач.