УДК 004.421+004.4’2+004.771

ПАРАЛЛЕЛЬНАЯ РЕАЛИЗАЦИЯ ЛОГИЧЕСКОГО МЕТОДА РЕШЕНИЯ ЗАДАЧ КАЧЕСТВЕННОГО АНАЛИЗА ДВОИЧНЫХ ДИНАМИЧЕСКИХ СИСТЕМ

Богданова Вера Геннадьевна
К.т.н., доцент, с.н.с., e-mail: bvg@icc.ru
Горский Сергей Алексеевич
К.т.н., н.с., e-mail: gorskysergey@mail.ru
Институт динамики систем и теории управления имени В.М. Матросова Сибирского отделения Российской академии наук (ИДСТУ СО РАН),
664033 г. Иркутск ул. Лермонтова, 134

Аннотация. Широкое применение двоичных динамических систем (ДДС) как в научных, так и прикладных исследованиях обуславливает актуальность разработки новых и совершенствования существующих методов качественного анализа поведения траекторий ДДС. Высокая вычислительная сложность этих задач требует разработки программных и инструментальных средств ее решения с использованием технологий параллельных и распределенных вычислений и обеспечением прозрачного доступа конечного пользователя к ресурсам высокопроизводительных вычислительных сред на основе сервис-ориентированного подхода. В настоящей работе рассматриваются параллельные программные средства реализации логического подхода к решению рассматриваемых задач, приводятся результаты вычислительных экспериментов.
Ключевые слова: двоичная динамическая система, булева модель, параллельный QBF решатель, качественный анализ.

Цитирование: Богданова В.Г., Горский С.А. Параллельная реализация логического метода решения задач качественного анализа двоичных динамических систем // Информационные и математические технологии в науке и управлении. 2018. №4 (12). С. 145–154. DOI: 10.25729/2413-0133-2018-4-15

Введение. Качественное исследование позволяет выявить существенные особенности динамической системы, важные как с теоретической, так и с практической точки зрения. Основной задачей качественного исследования является анализ поведения траекторий динамической системы с целью проверки, соответствует ли оно совокупности ограничений, характеризующих свойство. Для исследования двоичных динамических систем (ДДС) предлагается подход, основанный на методе булевых ограничений, применение которого для решения ряда задач представлено в работах [2, 8, 15]. На основе этого метода булева модель свойства ДДС записывается на языке булевых уравнений или булевых формул с кванторами. В настоящем докладе рассматриваются программные средства реализации логического метода, обеспечивающие построение булевых моделей описания свойств и проверку их выполнимости с помощью специализированных решателей булевых уравнений. Для качественного исследования ДДС в одном случае требуется решить задачу булевой выполнимости (SAT) или проверить истинность квантифицированной булевой формулы (QBF), в другом случае необходимо найти все решения булева уравнения. Все эти задачи

относятся к классу труднорешаемых [3]. Высокая вычислительная сложность SAT и QBF задач актуализирует необходимость разработки программных и инструментальных средств их решения с использованием параллельных и распределенных вычислений на основе сервис-ориентированных технологий для обеспечения прозрачного доступа конечного пользователя к ресурсам высокопроизводительных вычислительных сред.

Существующие программные средства качественного анализа автономных синхронных ДДС (называемых также булевыми сетями, Boolean Network) [10, 11, 13, 16] предназначены в основном только для поиска аттракторов. В работе [12] отмечается, что эти средства имеют ряд недостатков, в частности, ограничены сложностью булевой модели и форматом ее представления, требуют навыков программирования от предметных специалистов, поскольку часто используются только в качестве инструментов командной строки, зависящих от платформы.

Таким образом, при решении задач качественного анализа возникают следующие функциональных требования к разрабатываемому программному обеспечению: наличие средств построения булевой модели как для поиска аттракторов, так и для спецификации разнообразных динамических свойств ДДС (в частности, достижимости, изолированности, притяжения, связности) в требуемом формате и эффективных параллельных решателей SAT и TQBF задач. В рамках предлагаемого подхода этапы построения булевой модели, специфицирующей динамическое свойство и решения полученной системы булевых уравнений разделены и реализованы в виде независимых прикладных микросервисов. Для автоматизации создания микросервисов и организации управления их выполнением используются инструментальные средства [9, 15].

1. Построение булевой модели. Рассматривается автономная синхронная ДДС, векторно-матричное уравнение которой имеет вид

$$x^t = F(x^{t-1}), \qquad (1)$$

где x – вектор состояния, $x \in B^n$, $B = \{0,1\}$, n – размерность вектора состояния; $t \in T = \{1,2,...,k\}$ – дискретное время (номер такта); $F(x)$ – векторная функция алгебры логики, называемая функцией переходов.

Основные принципы технологии булева моделирования, в рамках которой описание булевой модели может быть задано как декларативным, так и процедурным путем, приведены в [5]. Основой построения булевой модели спецификации динамического свойства служит получаемая в соответствии с (1) булева формула одношагового перехода (левая часть уравнения $x^1 \oplus F(x^0) = 0$). Эта формула, в зависимости от конкретного свойства, используется для построения, например, *k*-перехода или циклической траектории длины *k*. ДДС может быть задана на математическом языке в виде (1), в формате Latex или MathML. Для совместимости с другими программными системами используется описание ДДС в формате «.cnet». В конечном итоге выходным форматом булевых ограничений для описания свойства являются DIMACS или QDIMACS форматы. Это универсальные форматы для SAT и TQBF задач, что позволяет выполнять этап решения с помощью соответствующих эффективных массивных параллельных решателей этих задач. В соответствии со способом описания ДДС подсистема булева моделирования включает сервисы конвертирования булевой формулы из одного формата в другой; а также сервис BFPDG (Boolean Function Procedural Description Generator) для генерирования процедурного описания булевой

функции по математическому описанию ДДС в виде (1) с последующим получением представления функции в дизъюнктивной (или конъюнктивной) нормальной форме (ДНФ или КНФ) с помощью построения таблицы истинности по этому процедурному описанию и преобразованием к требуемому выходному формату. В сервисе используется свободно распространяемый редактор FMath Javascript Equation Editor (http://www.fmath.info/html5-editor/use.jsp) (рис. 1). В зависимости от размерности n генерируется последовательное или

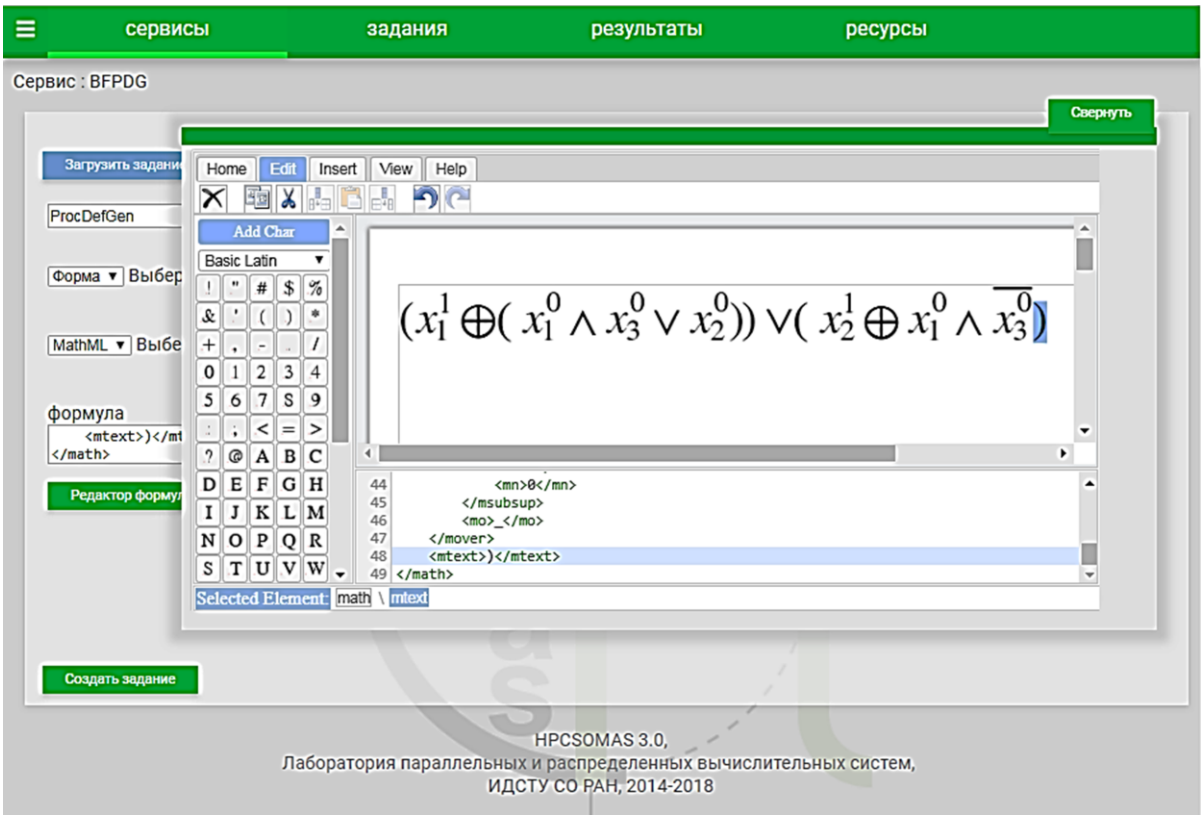

Рис. 1. Сервис генерации процедурного описания ДДС.

параллельное процедурное описание функции. Частный случай получения выходного формата DIMACS, когда левую часть уравнение $x^1 \oplus F(x^0) = 0$ можно представить в алгебраической нормальной форме (АНФ), описан в работе [2].

2. Подсистема поиска решения. Для параллельного поиска всех решений булева уравнения в формате DIMACS используется разработанный авторами решатель hpcsat [7] с использованием в качестве базового в дочерних процессах решателя инструментального комплекса РЕБУС [5]. Для решения задачи TQBF используется модифицированная версия решателя Hpcqsat [2].

Параллельный решатель Hpcqsat. Проверка выполнимости ряда свойств при качественном анализе ДДС (в частности, свойств связности и тотальной достижимости) на основе логического метода требует решения задачи проверки истинности квантифицированной булевой формулы (QBF). Разработанный авторами параллельный решатель таких задач ориентирован на частный случай этой задачи, 2QBF, в виде которой специфицируется описание проверяемого динамического свойства.

Подходы к параллельному решению проверки истинности QBF, основанные на модели общей и распределенной памяти, рассмотрены в работе [6]. В решателе Hpcqsat,

реализованном на основе архитектуры «Master-Slave», используется распараллеливание по данным с помощью расщепления исходной QBF. В дочерних процессах истинность остаточных формул определяется с помощью последовательного QBF решателя, называемого далее базовым решателем.

Параллельный решатель Hpcqsat представляет собой MPI-приложение на языке C++. Представленный ниже алгоритм, реализованный в Hpcqsat, описывает работу главного процесса (Master) и рабочих процессов (Slave). В главном процессе осуществляется работа с деревом расщепления исходной булевой модели (булевой функции) и управление работой дочерних процессов. В отличие от ранее описанного алгоритма [2], в новом варианте реализации работа главного процесса включает два этапа: расщепление булевой функции (Solve = 0) и управление решением полученных подзадач (Solve = 1). Такой подход, в сравнении с предыдущей версией, позволяет поддерживать количество подзадач в очереди в соответствии с количеством свободных вычислительных ресурсов, обеспечивая лучшую балансировку загрузки дочерних процессов. Описание алгоритма на псевдокоде:

```
//Master
FreeSubtask = 1; // Количество свободных подзадач
SubtaskCount = 1; // Общее количество подзадач
//FreeSubtask и SubtaskCount изменяются в процедурах Send, setSubtaskValue,
splitSubtask
Solve = 0; // 0 – расщепление; 1 – решение
main(){
|   while (true){
|   |   // Проверяем, есть ли свободные ресурсу и задачи
|   |   if (Requests > 0 and FreeSubtask > 0){
|   |   |   // Проверяем число подзадач
|   |   |   if (Solve == 0 and SubtaskCount < N){
|   |   |   |   if (findTaskForPrepare (&Task)) {Send(Task);}}
|   |   |   else{
|   |   |   |   Solve = 1;
|   |   |   |   // Ищем подзадачи для расщепления, которые не удалось решить
|   |   |   |   if (findNeedPrepare (&Task)) {Send(Task);}
|   |   |   |   // Если их нет ищем подзадачи для решения
|   |   |   |   else if (findTaskForSolve (&Task)) {Send(Task);}}}
|   |   else{
|   |   |   Recv (&result);
|   |   |   if (result.type == Request) {addToRequests (result);}
|   |   |   if (result.type == Solved) {
|   |   |   |   if (result.type == INDETERMINATE) {markTaskForSplit(result);}
|   |   |   |   else {setSubtaskValue(result);}}
|   |   |   if (result == Splited) {splitSubtask (result);}}
|   |   if (SubtaskCount == 0){return Solution;}}}

// Slave
main(){
|   while(true){
|   |   Send (Request); // Посылаем запрос на задание
|   |   Recv (&Task);
|   |   if (Task.type = Split){
|   |   |   Split (Task, &Result);
|   |   |   Send (Result);}
|   |   else {
|   |   |   Solve (Task);
|   |   |   Send (Result);}}}
```

Процедура выбора задач на расщепление `findTaskForPrepare` регулирует (по возможности, уравнивает) сложность подзадач, расщепляя в первую очередь остаточные булевы функции с наибольшим числом переменных, оставшихся после процедуры упрощения, входящей в процедуру анализа `Analyze` булевых функций подзадач, выполняемую в дочерних процессах. Процедура упрощения реализует метод распространения ограничений, в результате которого могут быть либо получены значения других переменных, либо решение данной подзадачи. В первом случае после завершения процесса упрощения в остаточной функции остаются не означенными k переменных, это число и принимается за сложность подзадачи. Именно из этих k переменных осуществляется выбор для дальнейшего расщепления данной подзадачи (если оно понадобится). Таким образом, при расщеплении задачи в главном процессе (рис. 2(а)) строится дерево расщепления, и листья этого дерева в общем случае будут получены означиванием разных переменных. На рис. 2(б) приведен пример построения дерева расщепления для квантифицированной булевой формулы $\forall x_1 x_2 x_3 x_4 \exists x_5 \dots x_{20} \phi(x_1, \dots, x_{20})$, где $\phi(x_1, \dots, x_{20})$ – булевы ограничения задачи.

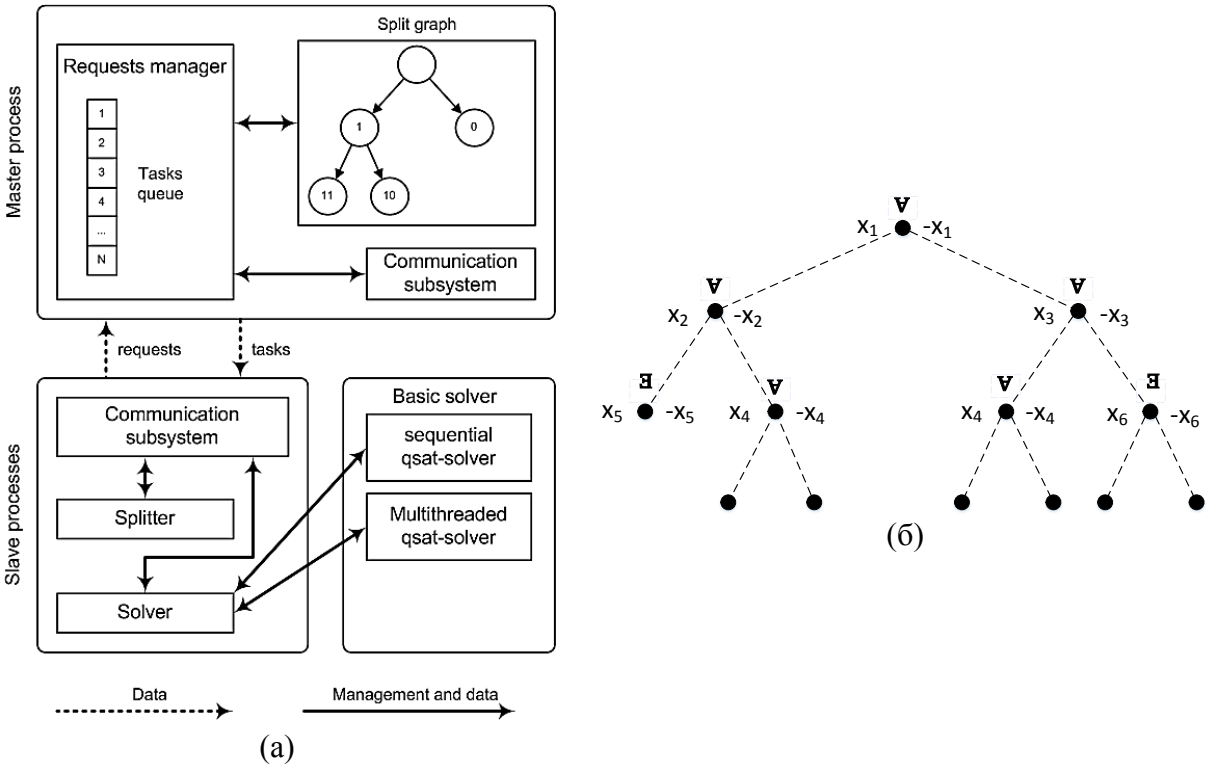

Рис. 2 (а). Архитектура решателя Hpcqsat; (б) дерево расщепления.

3. Вычислительные эксперименты. Цель эксперимента состояла, во-первых, в оценке характеристик решателя Hpcqsat в сравнении с аналогичным решателем HordeQBF [6]; во-вторых, в исследовании применимости нашего подхода в случае масштабируемого решения QBF задач. В рамках нашего исследования свободно распространяемые версии DepQBF [14] и HordeQBF установлены на кластере «Академик В.М. Матросов» для проведения сравнения параллельных решателей в одинаковых условиях. Оба сравниваемых решателя реализованы с использованием библиотеки MPI, в качестве базового решателя используют DepQBF. Отличия заключаются в следующем:

1) решатель HordeQBF основан на «портфолио» подходе, а Hpcqsat использует разделение поискового пространства;
2) Hpcqsat не использует, а HordeQBF применяет обмен конфликтными дизъюнктами;
3) Hpcqsat управляет очередью подзадач для дочерних процессов.

В табл. 1 приведены статистические данные результатов вычислительного эксперимента, проведенного с использованием ресурсов Иркутского суперкомпьютерного центра [4] для сравнения массивного параллельного решателя Hpcqsat с массивным параллельным решателем HordeQBF [6] на наборе тестовых 2QBF задач [1]. Ограничение на время решения одной задачи составляло 4800 секунд. Результаты вычислительного эксперимента показывают существенное преимущество Hpcqsat.

Таблица 1. Статистические результаты решения набора тестовых задач.

Процессорные ядра	Характеристика	Hpcqsat	HordeQBF
64	Общее время решения	2487,37	27798,72
	Среднее время решения	138,19	1544,37
	Количество нерешенных задач	0	2
256	Общее время решения	1049,75	26657,28
	Среднее время решения	58,32	1480,96
	Количество нерешенных задач	0	4
512	Общее время решения	855,86	29615,28
	Среднее время решения	47,55	1645,29
	Количество нерешенных задач	0	6

На рис. 3 приведены результаты среднего ускорения и эффективности решателя Hpcqsat на тестовых 2QBF с зафиксированной размерностью задачи при увеличении количества процессорных ядер. Ускорение скачкообразно возрастает. Соответственно скачкообразно ведет себя и эффективность. Наблюдается естественное снижение эффективности, т.к. размерность задачи не увеличивается. Тем не менее 40% значений лежат в приемлемом интервале от 0,5 до 1, т.е. для рассматриваемой размерности нужно подобрать соответствующее количество ядер для сохранения требуемого уровня эффективности.

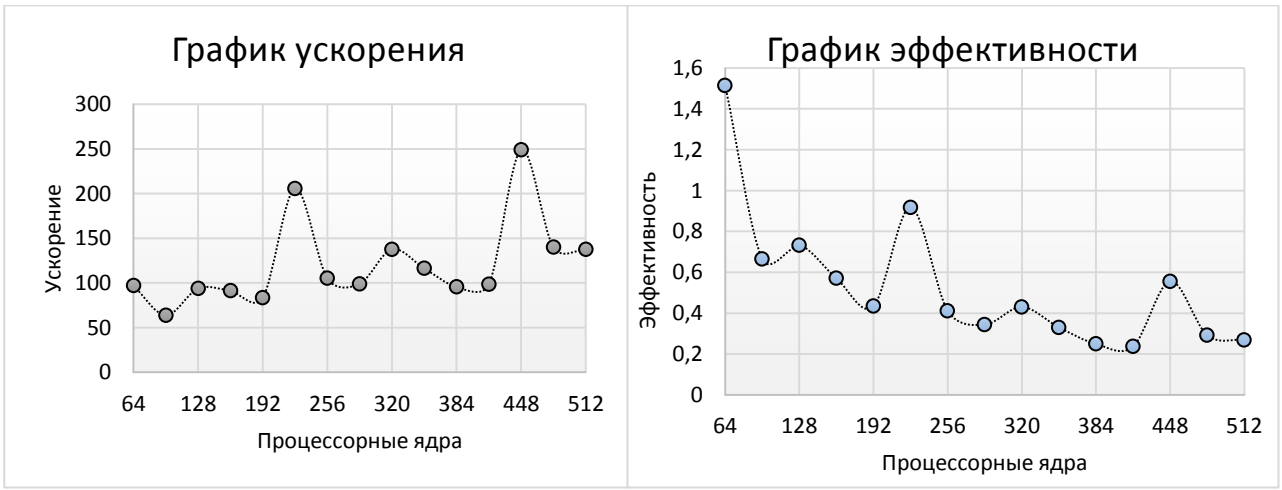

Рис. 3. Ускорение и эффективность решателя Hpcqsat.

Заключение. Выполнена параллельная программная реализация логического метода для решения задач качественного анализа ДДС. Автоматизированная технология булева моделирования, обеспечивающая описание ДДС в различных форматах, позволяет поддерживать совместимость с другими программными средствами качественного анализа. Проведены вычислительные эксперименты, подтверждающие существенное преимущество разработанного параллельного решателя Hpcqsat в сравнении с аналогичным решателем HordeQBF.

Исследование выполнено при поддержке РФФИ, проект № 18-07-00596/18.

СПИСОК ЛИТЕРАТУРЫ

1. Библиотека 2QBF задач. Режим доступа: www.qbflib.org/TS2010/2QBF.tar.gz.
2. Богданова В.Г., Горский С.А., Пашинин А.А. Web-сервис синтеза линейной обратной связи для двоичных динамических систем // Информационные и математические технологии в науке и управлении. 2017. № 4. С. 62–70.
3. Гэри М., Джонсон Д. Вычислительные машины и труднорешаемые задачи. М.: Мир. 1982. 416 с.
4. Иркутский суперкомпьютерный центр. Режим доступа: http://hpc.icc.ru/ [online, accessed: 31-May-2018].
5. Опарин Г.А., Богданова В.Г. РЕБУС - интеллектуальный решатель комбинаторных задач в булевых ограничениях // Вестник Новосибирского государственного университета. Серия: Информационные технологии. 2008. Т. 6. № 1. С. 61–69.
6. Balyo T., Lonsing F. HordeQBF: A Modular and Massively Parallel QBF Solver // Proceedings of 19th International Conference, Bordeaux, France, July 5-8, LNCS. 2016. Vol. 9710. Pp. 531–538. DOI 10.1007/978-3-319-40970-2_33.
7. Bogdanova V.G., Gorsky S.A. Scalable parallel solver of Boolean satisfiability problems // 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO). Opatija. 2018. Pp. 0222–0227. doi: 10.23919/MIPRO.2018.8400042.
8. Bychkov I.V., Oparin G.A., Bogdanova V.G., Pashinin A.A. The Applied Problems Solving Technology Based on Distributed Computational Subject Domain Model: a Decentralized Approach // Параллельные вычислительные технологии – XII международная конференция, ПаВТ'2018, г. Ростов-на-Дону, 2–6 апреля 2018 г. Короткие статьи и описания плакатов. Челябинск: Издательский центр ЮУрГУ. 2018. С. 34–48.
9. Bychkov I.V., Oparin G.A., Bogdanova V.G., Pashinin A.A., Gorsky S.A. Automation Development Framework of Scalable Scientific Web Applications Based on Subject Domain Knowledge. In: Malyshkin V. (eds) Parallel Computing Technologies. PaCT 2017. Lecture Notes in Computer Science. 2017. Vol. 10421. Pp. 278–288. Springer, Cham.
10. Dubrova E., Teslenko M. A SAT-based algorithm for finding attractors in synchronous Boolean networks // IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB) Volume 8. Issue 5. September 2011. Pp. 1393–1399. doi: 10.1109/TCBB.2010.20.

11. Dubrova E., Teslenko M., Martinelli A. Kauffman Networks: Analysis and Applications // Proceedings of International Conference on Computer-Aided Design (ICCAD'2005). November 6-10, 2005. San Jose, CA, USA. Pp. 479–484.
12. Guo W., Yang G., Wu W., He L., Sun M. A Parallel Attractor Finding Algorithm Based on Boolean Satisfiability for Genetic Regulatory Networks. PloS one. 2014. DOI: 10.1371/journal.pone.0094258.
13. He Z., Zhan M., Liu S., Fang Z., Yao C. An Algorithm for Finding the Singleton Attractors and Pre-Images in Strong-Inhibition Boolean Networks. PLoS ONE 11(11): e0166906, 2016. doi:10.1371/journal.pone.0166906.
14. Lonsing F., Biere A. DepQBF: A Dependency-Aware QBF Solver // Journal of Satisfiability, Boolean Modeling and Computation. 2010. Vol. 9. Pp. 71–76.
15. Oparin G. A., Bogdanova V. G., Pashinin A. A., Gorsky S. A. Distributed solvers of applied problems based on microservices and agent networks // 2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO). Opatija. 2018. Pp. 1415–1420. doi: 10.23919/MIPRO.2018.8400255.
16. Zheng D., Yang G., Li X., Wang Z., Hung W.N. An efficient algorithm for finding attractors in synchronous Boolean networks with biochemical applications // Genetics and Molecular Research. 2013. 12 (4). Pp. 4656–4666. doi: 10.4238/2013.October.18.4.

UDK 004.421+004.4’2+004.771

PARALLEL IMPLEMENTATION OF THE LOGICAL METHOD FOR SOLVING THE PROBLEMS OF BINARY DYNAMIC SYSTEMS QUALITATIVE ANALYSIS

Vera G. Bogdanova
Ph. D., assistant professor, senior researcher, e-mail: bvg@icc.ru
Sergey A. Gorsky
Ph. D., researcher, e-mail: gorskysergey@mail.ru
Matrosov Institute for System Dynamics and Control Theory Siberian Branch of the Russian Academy of Sciences, 134, Lermontov Str., 664033, Irkutsk, Russia

Abstract. The wide application of binary dynamic systems (BDS) in both scientific and applied research causes the urgency of developing new and improving existing methods for qualitative analysis of the behavior of DDS trajectories. High computational complexity of these tasks requires the development of software and tools for its solution using parallel and distributed computing technologies and service-oriented access to the resources of high-performance computing environments. In this paper, parallel software tools for implementing a logical approach to solving the problems under consideration are considered, the results of computational experiments.

Keywords: binary dynamic system, Boolean model, parallel QBF solver, qualitative analysis.

References

1. Biblioteka 2QBF zadach [Library of 2qbf] Available at: www.qbflib.org/TS2010/2QBF.tar.gz. (accessed 3105.2018) (in Russian)
2. Bogdanova V.G., Gorsky S.A., Pashinin A. A. Web-servis sinteza linejnoj obratnoj svyazi dlya dvoichnyh dinamicheskih system [Web-service of linear feedback synthesis for binary dynamic systems] // Informatsionnyye i matematicheskiye tekhnologii v nauke i menedzhmente = Information and mathematical technologies in science and management. 2017. № 4. Pp. 62–70. (in Russian)
3. Gary M., Johnson D. Vychislitel'nye mashiny i trudnoreshaemye zadachi [Computers and Intractability: A Guide to the Theory of NP-Completeness]. Moscow. Mir. 1982. 416 p. (in Russian)
4. Irkutskiy superkomp'yuternyy tsentr [Irkutsk Supercomputer Center of SB RAS]. Available at: http://hpc.icc.ru/ (accessed 31-May-2018) (in Russian)
5. Oparin G.A., Bogdanova V.G. REBUS - intellektual'nyj reshatel' kombinatornyh zadach v bulevyh ogranicheniyah [Rebus – Intellctual Solver for Combinatorial Problems in Boolean Constraints] // Vestnik Novosibirskogo gosudarstvennogo universiteta. Seriya: Informacionnye tekhnologii = Novosibirsk State University Journal of Information Technologies. 2008. T. 6. № 1. Pp. 61–69. (in Russian)
6. Balyo T., Lonsing F. HordeQBF: A Modular and Massively Parallel QBF Solver // Proceedings of 19th International Conference, Bordeaux, France, July 5-8, LNCS. 2016. Vol. 9710. Pp. 531–538. DOI 10.1007/978-3-319-40970-2_33.
7. Bogdanova V.G., Gorsky S.A. Scalable parallel solver of Boolean satisfiability problems // 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO). Opatija. 2018. Pp. 0222–0227. doi: 10.23919/MIPRO.2018.8400042.
8. Bychkov I.V., Oparin G.A., Bogdanova V.G., Pashinin A.A. The Applied Problems Solving Technology Based on Distributed Computational Subject Domain Model: a Decentralized Approach // Параллельные вычислительные технологии – XII международная конференция, ПаВТ'2018, г. Ростов-на-Дону, 2–6 апреля 2018 г. Короткие статьи и описания плакатов. Челябинск: Издательский центр ЮУрГУ. 2018. С. 34–48.
9. Bychkov I.V., Oparin G.A., Bogdanova V.G., Pashinin A.A., Gorsky S.A. Automation Development Framework of Scalable Scientific Web Applications Based on Subject Domain Knowledge. In: Malyshkin V. (eds) Parallel Computing Technologies. PaCT 2017. Lecture Notes in Computer Science. 2017. Vol. 10421. Pp. 278–288. Springer, Cham.
10. Dubrova E., Teslenko M. A SAT-based algorithm for finding attractors in synchronous Boolean networks // IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB) Volume 8. Issue 5. September 2011. Pp. 1393–1399. doi: 10.1109/TCBB.2010.20.
11. Dubrova E., Teslenko M., Martinelli A. Kauffman Networks: Analysis and Applications // Proceedings of International Conference on Computer-Aided Design (ICCAD'2005). November 6-10, 2005. San Jose, CA, USA. Pp. 479–484.
12. Guo W., Yang G., Wu W., He L., Sun M. A Parallel Attractor Finding Algorithm Based on Boolean Satisfiability for Genetic Regulatory Networks. PloS one. 2014. DOI: 10.1371/journal.pone.0094258.

13. He Z., Zhan M., Liu S., Fang Z., Yao C. An Algorithm for Finding the Singleton Attractors and Pre-Images in Strong-Inhibition Boolean Networks. PLoS ONE 11(11): e0166906, 2016. doi:10.1371/journal.pone.0166906.
14. Lonsing F., Biere A. DepQBF: A Dependency-Aware QBF Solver // Journal of Satisfiability, Boolean Modeling and Computation. 2010. Vol. 9. Pp. 71–76.
15. Oparin G. A., Bogdanova V. G., Pashinin A. A., Gorsky S. A. Distributed solvers of applied problems based on microservices and agent networks // 2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO). Opatija. 2018. Pp. 1415–1420. doi: 10.23919/MIPRO.2018.8400255.
16. Zheng D., Yang G., Li X., Wang Z., Hung W.N. An efficient algorithm for finding attractors in synchronous Boolean networks with biochemical applications // Genetics and Molecular Research. 2013. 12 (4). Pp. 4656–4666. doi: 10.4238/2013.October.18.4.