Серия «Математика»
2026. Т. 55. С. 94—109

Онлайн-доступ к журналу:
http://mathizv.isu.ru

И З В Е С Т И Я
Иркутского государственного университета

Научная статья

УДК 510.643
MSC 03B44, 03B42, 03A05, 03B45, 03B70
DOI https://doi.org/10.26516/1997-7670.2026.55.94

Разрешимость многоагентной логики деревьев вычислений $\mathcal{CTLK}^{Rel}$

С. И. Башмаков[1✉], **К. А. Смелых**[1]

[1] Сибирский федеральный университет, Красноярск, Российская Федерация
✉ krauder@mail.ru

Аннотация: Исследуется многоагентная логика деревьев вычислений относительно реляционной семантики возможных миров Крипке: вопросы разрешимости логики, сложности построения моделей, проверки выполнимости, корректности. Для введённой ранее семантики доказано строгое свойство конечной модели, получены полиномиальные оценки размерности минимальных моделей для произвольных формул. Доказана рекурсивная перечислимость конечных фреймов логики, предложен эффективный алгоритм проверки выполнимости формул, позволяющий заключить разрешимость логики. Полученные полиномиальные оценки находятся в рамках теоретических ожиданий, что позволяет воспринимать исследуемую логику эффективным инструментом анализа многоагентных распределённых систем и практического model-checking, рассчитывать на положительное разрешение вопросов унификации и описания допустимости правил вывода.

Ключевые слова: многоагентная логика, ветвящаяся временная логика, реляционная семантика Крипке, разрешимость, метод построения модели

Благодарности: Работа поддержана Красноярским математическим центром, финансируемым Минобрнауки РФ (Соглашение 075-02-2025-1790).

Ссылка для цитирования: Башмаков С. И., Смелых К. А. Разрешимость многоагентной логики деревьев вычислений $\mathcal{CTLK}^{Rel}$ // Известия Иркутского государственного университета. Серия Математика. 2026. Т. 55. С. 94–109.
https://doi.org/10.26516/1997-7670.2026.55.94

Research article

Decidability of Multi-agent Logic of Computation Trees $\mathcal{CTLK}^{Rel}$

Stepan I. Bashmakov[1]✉, **Kirill A. Smelykh**[1]

[1] Siberian Federal University, Krasnoyarsk, Russian Federation
✉ krauder@mail.ru

Abstract: We continue to explore the multi-agent logic of computational trees relative to the relational Kripke semantics of possible worlds: we investigate the question of logical solvability, the complexity of model construction, feasibility testing, and correctness.
For the semantics introduced earlier, we proved a strong finite model property, and obtained polynomial estimates of the dimension of minimal models for an arbitrary formulas. We proved the recursive enumerability of finite frames of logic and proposed an effective algorithm for checking the feasibility of formulas, which makes it possible to conclude the solvability of logic. The obtained polynomial estimates are within the framework of theoretical expectations, which makes it possible to perceive the logic under study as an effective tool for analyzing multi-agent distributed systems and practical model-checking, and to count on a positive resolution of issues of unification and description of the admissibility for inference rules.

Keywords: multi-agent logic, branching temporal logic, Kripke relational semantics, decidability, model construction method

Acknowledgements: This work was supported by the Krasnoyarsk Mathematical Center funded by the Ministry of Education and Science of the Russian Federation (Agreement 075-02-2025-1790).

For citation: Bashmakov S. I., Smelykh K. A. Decidability of Multi-agent Logic of Computation Trees $\mathcal{CTLK}^{Rel}$. *The Bulletin of Irkutsk State University. Series Mathematics*, 2026, vol. 55, pp. 94–109. (in Russian)
https://doi.org/10.26516/1997-7670.2026.55.94

1. Введение

Современные исследования в области вычислительных систем и мультиагентных систем (MAS) сталкиваются с множеством вызовов, связанных с описанием, анализом и верификацией сложных процессов, возникающих в распределённых вычислениях [1]. Мультиагентные системы находят применение в самых разных областях знания, включая робототехнику, интеллектуальные транспортные системы, управление энергоресурсами и моделирование социального поведения [4]. В таких системах взаимодействие агентов играет центральную роль, определяя эффективность работы всей системы. Для обеспечения надёжности и прогнозируемости таких систем критически важно использование формальных методов, включая логические формализмы [11].

Особую роль в этом контексте играет логика деревьев вычислений $\mathcal{CTL}$ и её многопользовательские расширения, к примеру исследуемая нами $\mathcal{CTLK}$ (Computation Tree Logic with Knowledge). Эти формализмы предоставляют эффективный аппарат для описания как временных аспектов поведения систем, так и доступной агентам информации. Однако классические подходы к семантике этих логик не всегда позволяют учитывать сложные взаимосвязи между агентами и их взаимодействие в условиях динамически изменяющихся сред. Реляционная семантика логики $\mathcal{CTLK}^{Rel}$ решает эту проблему, реализуя возможность детализированного описания временных и модальных отношений.

Многоагентные системы, как правило, функционируют в распределённых средах, где состояние определяется взаимодействием множества независимых участников. Это создаёт дополнительные сложности для анализа, такие как необходимость проверки глобальных свойств системы на основе локальных данных. В таких условиях важным инструментом становится проверка моделей (model-checking) — формальный метод, позволяющий автоматически определять, удовлетворяет ли система заданным требованиям. Применение логик $\mathcal{CTLK}$ и $\mathcal{CTLK}^{Rel}$ в model-checking предоставляет мощные возможности для исследования таких свойств, как безопасность, справедливость и достижимость.

Реляционная семантика логики $\mathcal{CTLK}^{Rel}$, в отличие от классических подходов в виде маркированной системы переходов (LTS) или сетей Петри [9], позволяет моделировать иерархические и древовидные структуры взаимодействий агентов. Это делает её особенно полезной для анализа систем, где присутствуют как кооперативные, так и конкурентные отношения между агентами. Например, в интеллектуальных транспортных системах отдельные транспортные средства могут координировать свои действия для оптимизации движения, при этом сохраняя индивидуальные цели. Аналогичным образом в системах управления энергией агенты могут представлять различные узлы энергосети, стремящиеся минимизировать локальные затраты, одновременно обеспечивая глобальный баланс.

Настоящая работа посвящена исследованию разрешимости логики $\mathcal{CTLK}^{Rel}$ [1] и описанию алгоритмов построения моделей. Также проводится оценка алгоритмической сложности предложенных методов.

2. Основные определения

Прежде чем ввести специальные обозначения, дадим ключевые определения теории модальных логик, используемые далее [5]. Под языком модальной логики $L^{\mathcal{ML}}$ мы будем понимать совокупность пропозициональных переменных $Prop$, скобки $(,)$, стандартные булевы операции $\vee, \wedge, \rightarrow, \neg$ и одноместный модальный оператор общезначимости $\Box$.

Выражение в языке $L^{\mathcal{ML}}$ назовём правильно построенной формулой (сокращено п. п. ф.), если оно построено по следующим правилам:

– $p \in Prop$ является п. п. ф.;
– если ϕ и ψ — п. п. ф., то $\phi \wedge \psi$, $\phi \to \psi$, $\phi \vee \psi$, $\neg\phi$ — п. п. ф.;
– применение модального оператора $\Box$ к п. п. ф. образует п. п. ф.

Оператор возможности $\Diamond$ выразим в языке стандартным образом: $\Diamond\varphi = \neg\Box\neg\varphi$, константа $\bot$ — выражением $\bot := p \wedge \neg p$. Множество всех правильно построенных формул в языке $L^{\mathcal{ML}}$ обозначим $For(L^{\mathcal{ML}})$.

Модальным фреймом Крипке F называется пара $F = \langle W, R\rangle$, состоящая из непустого множества W и бинарного отношения R на W. Элементы W называются мирами или состояниями. Если xRy, мы говорим, что y достижим из x или x видит y.

Определение 1. *Логика $\mathcal{ML}$ называется финитно аппроксимируемой (или обладает свойством конечной модели), если существует класс конечных фреймов C такой, что*

$$\mathcal{ML} = \{\varphi \in For(L^{\mathcal{ML}}) : \ \forall F \in C \ F \Vdash \varphi\}.$$

Определение 2. *Формула φ находится в отрицательной нормальной форме (NNF), если операторы отрицания стоят только перед пропозициональными переменными.*

3. Реляционная семантика логики $L^{\mathcal{CTLK}^{Rel}}$

Основные определения и обозначения реляционной семантики логики $\mathcal{CTLK}^{Rel}$ даны в [1]. Приведём основные из них.

Алфавит языка $L^{\mathcal{CTLK}}$ включает счетный набор пропозициональных переменных $Prop = \{p_1, \ldots, p_n, \ldots\}$, скобки $(,)$ стандартные булевы операции, временные операторы $\{AX, EX, AU, EU\}$ и множество одноместных модальных операторов $\{\Box_{\leq}, \Box_1, \ldots, \Box_n, ...\}$.

Определение 3. *Выражение в языке $L^{\mathcal{CTLK}}$ называется правильно построенной формулой (сокращено п. п. ф.), если оно построено по следующим правилам:*

1) $p \in Prop$ является п. п. ф.;

2) если ϕ и ψ — п. п. ф., то $\phi \wedge \psi$, $\phi \to \psi$, $\phi \vee \psi$, $\neg\phi$ — п. п. ф.;

3) применение одноместных модальных операторов $\Box_{\leq}, \Box_1, \ldots,$ $\Box_n, \ldots$ к п. п. ф. образует п. п. ф.;

4) применение одноместных временных операторов EX и AX к п. п. ф. образует п. п. ф.;

5) если φ и ψ — п. п. ф. то $A(\varphi U \psi)$ и $E(\varphi U \psi)$ — п. п. ф.

Множество правильно построенных формул в языке $L^{\mathcal{CTLK}}$ будем обозначать $For(L^{\mathcal{CTLK}})$.

3.1. Семантика Крипке для логики $\mathcal{CTLK}^{Rel}$

Определение 4. *$\mathcal{CTLK}^{Rel}$-фреймом назовём упорядоченный набор $F := \langle W, R_{\leq}, R_1, \dots, R_n, \dots \rangle$, где*

1) W — объединение непустых непересекающихся множеств временных слоев:

$$W = \bigcup_{t \in \mathbb{N}} C_t,\ C_{t_1} \cap C_{t_2} = \emptyset, \text{ если } t_1 \neq t_2;$$

2) R_i, где $i \in \mathbb{N}$ — линейное отношение порядка, задающее последовательность состояний:

$$\forall a, b \in W : aR_ib \Rightarrow \exists k, j \in \mathbb{N}\, (k \leq j)\ (a \in C_k)\&(b \in C_j)$$

$$\&\, (k = j \iff a = b);$$

3) $R_{\leq}$ — отношение частичного порядка на элементах W, задающее временную связь между слоями:

$$\bigcup_{\forall i \in \mathcal{I}} R_i = R_{\leq};\ \ \forall b \in W \exists a \in C_0 : aR_{\leq}b.$$

Модель на $\mathcal{CTLK}^{Rel}$-фрейме F определим как пару $M := \langle F, V \rangle$, где $V : Prop \mapsto 2^W$. Тогда $\forall a \in C_t \subseteq W$, $\forall t \in \mathbb{N}$:
– $\langle M, a \rangle \Vdash_V \Box_{\leq}\varphi \Leftrightarrow \forall z \in W : aR_{\leq}z \Rightarrow \langle M, z \rangle \Vdash_V \varphi$;
– $\forall i \in \mathcal{I}\, \langle M, a \rangle \Vdash_V \Box_i\varphi \Leftrightarrow \forall z \in W : aR_iz \Rightarrow \langle M, z \rangle \Vdash_V \varphi$;
– $\langle M, a \rangle \Vdash_V EX\varphi \Leftrightarrow \exists b \in C_{t+1} :\Rightarrow b \Vdash_V \varphi$;
– $\langle M, a \rangle \Vdash_V AX\varphi \Leftrightarrow \forall b \in C_{t+1} :\Rightarrow b \Vdash_V \varphi$;
– $\langle M, a \rangle \Vdash_V A(\varphi\, U\, \psi) \Leftrightarrow (\forall i \in \mathcal{I})(\exists j \in \mathbb{N})\exists b \in C_{t+j} : aR_ib : b \Vdash_V \psi$ и $\forall c \in (a, \dots, a_{j-1})^i$, где $a_j = b$ и $c \Vdash_V \varphi$;
– $\langle M, a \rangle \Vdash_V E(\varphi\, U\, \psi) \Leftrightarrow (\exists i \in \mathcal{I})(\exists j \in \mathbb{N})\exists b \in C_{t+j} : aR_ib : b \Vdash_V \psi$ и $\forall c \in (a, \dots, a_{j-1})^i$, где $a_j = b$ и $c \Vdash_V \varphi$.

Фрейм интерпретируется как вычислительное дерево: каждая ветвь соответствует альтернативному ходу вычислений. Реляционная модель объединяет все такие альтернативы и потому может быть как конечной, так и бесконечной, — это зависит от того, остановится ли рассматриваемый процесс. При появлении новой ветви вводится отдельный агент, «ответственный» за свой маршрут. Узлы дерева сгруппированы по временным слоям; в рамках одного слоя расположены состояния, существующие в один и тот же момент. Бинарное отношение достижимости задаёт, к каким состояниям можно перейти из данного, когда время продвигается вперёд.

Определение 5. *Логикой $\mathcal{CTLK}^{Rel}$ языка $L^{\mathcal{CTLK}}$ будем называть множество всех $\mathcal{CTLK}$-формул, истинных на множестве всех фреймов некоторого ограниченного класса C:*

$$\mathcal{CTLK}^{Rel} = \{\varphi \in For(L^{\mathcal{CTLK}}) : F \Vdash \varphi \forall F \in C\}.$$

4. Разрешимость

Логика называется разрешимой, если существует алгоритм, который может определить, принадлежит ли данная формула логике или нет.

Доказательство разрешимости логики может следовать двум подходам: *семантическим* (через соответствующее определение логики на некотором классе моделей) или *синтаксическим* (в случае если логика определена через набор аксиом). К примеру, в [3] приводятся две таких стратегии. Однако, независимо от выбора, полезным первым шагом на пути доказательства разрешимости является подтверждение свойства конечной модели, установленное нами ранее [1].

При выборе *семантического* подхода для доказательства разрешимости нам потребуется установить два факта [3, Теорема 6.7]:
- сильное свойство конечной модели;
- рекурсивно перечислимый класс конечных фреймов.

Для начала выделим грубую оценку разрешимости. В силу доказательства финитной аппроксимируемости методом фильтрации [1], мы можем определить, что для любой формулы φ существует вычислимая функция f (метод фильтрации) такая, что $f(|\varphi|)$ является верхней границей размера этих моделей, необходимых для выполнимости φ. Воспользуемся следующим определением [3, Определение 6.6].

Определение 6. *(Строгое свойство конечной модели.) Пусть $\mathcal{ML}$ — нормальная модальная логика, M — множество конечных моделей таких, что $\mathcal{ML} = \mathcal{ML}_M$ и f — функция отображения на натуральные числа. $\mathcal{ML}$ обладает свойством $f(n)$-размерности модели относительно M, если каждая непротиворечивая $\varphi \in \mathcal{ML}$ формула удовлетворяет модели M, содержащей не более $f(|\varphi|)$ состояний.*

Однако для оценки верхней границы необходимо рассматривать формулу φ как множество, состоящее из подформул формулы φ.

Определение 7. *Замыкание формулы φ (обознач. $\Sigma(\varphi)$) — это множество, замкнутое относительно подформул φ. Описать замыкание формулы φ можно следующей схемой:*
- *если ψ —пропозициональная переменная p, то $\Sigma(\psi) = \{\psi\}$;*
- *если $\psi = \neg\chi$, то $\Sigma(\psi) = \{\psi\} \cup \Sigma(\chi)$;*
- *если $\psi = \chi_1 \wedge \chi_2$ или $\psi = \chi_1 \vee \chi_2$, то $\Sigma(\psi) = \{\psi\} \cup \Sigma(\chi_1) \cup \Sigma(\chi_2)$;*
- *если $\psi = EX\chi$ или $\psi = AX\chi$, то $\Sigma(\psi) = \{\psi\} \cup \Sigma(\chi)$;*

– *если* $\psi = E[\chi_1 U \chi_2]$ *или* $\psi = A[\chi_1 U \chi_2]$, *то* $\Sigma(\psi) = \{\psi\} \cup \Sigma(\chi_1) \cup \Sigma(\chi_2)$;
– *если* $\psi = \Box_i \chi$ *или* $\psi = \Diamond_i \chi$, *то* $\Sigma(\psi) = \{\psi\} \cup \Sigma(\chi)$;
– *если* $\psi = \Box_{\leq} \chi$, *то* $\Sigma(\psi) = \{\psi\} \cup \Sigma(\chi)$.

Выделим множество значимых подформул.

Определение 8. *Максимальным согласованным множеством для формулы* φ *называется подмножество* $\Gamma \subseteq \Sigma(\varphi)$, *причем:*

1) Γ *согласовано, если найдётся* $\psi \in \Sigma(\varphi)$, *такой что* $\psi \in \Gamma$ *и* $\neg\psi \in \Gamma$;
2) Γ *максимально, если для любой* $\psi \in \Sigma(\varphi)$, *либо* $\psi \in \Gamma$, *либо* $\neg\psi \in \Gamma$.

Результаты [3] фактически устанавливают сильное свойство конечной модели. Имеет место следующая

Теорема 1. *Логика* $\mathcal{CTLK}^{Rel}$ *обладает строгим свойством конечной модели: для любой выполнимой на бесконечной модели логики* $\mathcal{CTLK}^{Rel}$ *формулы* φ *существует конечная модель* $\mathcal{M}$, *на которой* φ *выполнима. Кроме того, число состояний в этой модели не превышает* $2^{|\Gamma|}$.

Доказательство. Согласно определениями 7 и 8, Γ содержит все значимые для фильтрации подформулы φ и $|\Gamma| \leq |\Sigma(\varphi)|$. Рассмотрим модель $\mathcal{M}$ на $\mathcal{CTLK}^{Rel}$-фрейме F. Определим отношение эквивалентности $\sim$ на множестве состояний W:

$$x \sim y \iff \forall \psi \in \Sigma(\varphi): \ \mathcal{M}, x \models \psi \Leftrightarrow \mathcal{M}, y \models \psi.$$

Иначе, два состояния эквивалентны, если удовлетворяют одним и тем же подформулам из Γ. Тогда W' — множество всех классов эквивалентности, $V'(p) = \{[x] \,|\, x \in V(p)\}$. Определим отношение на модели:

$$S_i = \{([x],[y]) \mid \exists x' \in [x],\, y' \in [y] : \, x' R_i y'\}, S_{\leq} = \bigcup_{\forall i \in \mathcal{I}} S_i.$$

Далее восстанавливаем свойства фрейма, используя транзитивное замыкание отношения:

$$\widehat{S_i} = \{\langle [x],[y] \rangle : \exists n > 0 [x] S_i^n [y]\}, \qquad \bigcup_{\forall i \in \mathcal{I}} \widehat{S_i} = \widehat{S}_{\leq}.$$

Таким образом, модель $M' = \langle W', \widehat{S_{\leq}}, \widehat{S_1}, \ldots, \widehat{S_n}, \ldots \rangle$ удовлетворяет всем свойствам отношений в $\mathcal{CTLK}^{Rel}$, в силу доказанного в [1], истинность на полученной модели сохраняется.

Мощность множества W' всех классов эквивалентности, полученных таким образом, не превышает числа всех возможных наборов истинностных значений формул из Γ (каждая формула может быть истинной или ложной), поэтому $|W'| \leq 2^{|\Gamma|}$.

Построенная модель $\mathcal{M}'$ является конечной, в ней выполняется формула φ, и количество её состояний не превышает $2^{|\Gamma|}$. Следовательно, в силу произвольности выбора формулы φ, логика $\mathcal{CTLK}^{Rel}$ обладает строгим свойством конечной модели, где любая выполнимая формула φ имеет конечную модель размера не более $2^{|\Gamma|}$. $\square$

Необходимо проверить, что соответствующий набор конечных моделей рекурсивно перечислим. Как отмечают в [5, Лемма 16.12], любой конечный граф/матрицу/фрейм можно однозначно закодировать конечной бинарной строкой; перебирая все такие строки и отбрасывая некорректные, получаем рекурсивную нумерацию нужного класса структур. Конкретно, каждый $\mathcal{CTLK}^{Rel}$-фрейм с n состояниями кодируем строкой $\langle n, R_{\leq}, R_1, \ldots, R_k\rangle$, где $R_{\leq}$ и R_i представлены $n \times n$-матрицами нулей-единиц. За конечное время проверяются тесты:

– рефлексивность и транзитивность $R_{\leq}$;
– вложения $R_i \subseteq R_{\leq}$ и линейность каждого R_i.

Перебор всех бинарных строк и отбрасывание некорректных по этим тестам порождает именно корректные конечные фреймы, следовательно указанный класс рекурсивно перечислим. Вместе с полученной ранее вычислимой границей $2^{|\Gamma|}$ на размер фильтрованной модели, это даёт разрешимость $\mathcal{CTLK}^{Rel}$ за время $O(2^{|\Gamma|})$.

Однако мы можем улучшить данную оценку. Реализуем алгоритм построения модели. Для этого используем тот факт, что формула φ общезначима тогда и только тогда, когда формула $\neg\varphi$ невыполнима, следовательно невозможно построить модель, удовлетворяющую этой формуле. Таким образом, задача разрешимости сводится к задаче проверки возможности построить модель, на которой формула $\neg\varphi$ невыполнима.

5. Построение модели

Вначале реализуем построение *абстрактного синтаксического дерева* (AST) нашей формулы. В каждый узел положим оператор, который должен быть соединен с подформулой, к которой применяется, связью. AST назовём построенным, когда на конечных узлах останутся только $p \in Prop$. Определим $unary\_op \in \{AX, EX, \Box_i, \Box_{\leq}, \Diamond_i, \Diamond_{\leq}\}$ и $binary\_op \in \{AU, EU, \wedge, \vee\}$. Опишем алгоритм построения AST формулы φ.

1) **Пропозициональная переменная.**

   Если строка φ соответствует переменной, создать листовой узел с меткой φ и вернуть его.

2) **Унарный оператор.**

   Если φ распадается как

$$\varphi = unary\_\,op\,\psi,$$

то создать узел с меткой $unary\_\,op$, рекурсивно вызвать

$$node.child = BuildAST(\psi)$$

и вернуть этот узел.

3) **Бинарный оператор.**

Если φ имеет форму

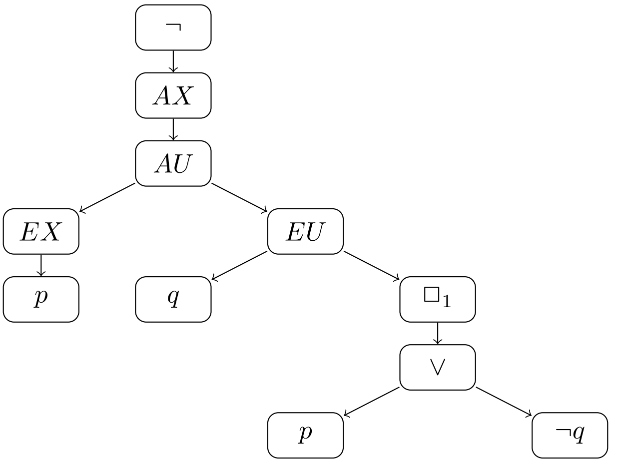
$$\varphi = (\psi_1\ binary\_\,op\,\psi_2),$$

то выделить подстроки ψ_1 и ψ_2, создать узел с меткой $binary\_\,op$, затем рекурсивно заполнить

$$node.left = BuildAST(\psi_1), \quad node.right = BuildAST(\psi_2)$$

и вернуть этот узел.

Пример 1. Пусть $\varphi = AXA(EXp\,U\,E(q\,U\,\Box_1(p\vee\neg q)))$. Тогда AST для неё будет выглядеть следующим образом:



Далее нам необходимо привести формулу в *отрицательную нормальную форму* (сокр. NNF). Для этого определим правила раскрытия отрицаний [8]:

1) $\neg AX\alpha = EX\neg\alpha$;
2) $\neg\Box_i\alpha = \Diamond_i\neg\alpha$;
3) $\neg\Box_{\leq}\alpha = \Diamond_{\leq}\neg\alpha$;
4) $\neg(A(\alpha\,U\,\beta) = (\neg\alpha\wedge\neg\beta\vee(\neg\alpha\wedge EX\neg(A(\alpha U\beta)))$;
5) $\neg(E(\alpha\,U\,\beta) = (\neg\alpha\wedge\neg\beta\vee(\neg\alpha\wedge AX\neg(A(\alpha U\beta)))$;

6) $\neg(\alpha \wedge \beta) = (\neg\alpha \vee \neg\beta)$;
7) $\neg(\alpha \vee \beta) = (\neg\alpha \wedge \neg\beta)$;
8) $\neg\neg\alpha = \alpha$.

Опишем алгоритм преобразования AST в NNF [6], согласно указанному выше соответствию:

– **Узел** $\boldsymbol{NOT}$**.**
  Пусть c — единственный потомок текущего узла.
  Если $c.type = NOT$, то
  $$ToNNF(node) = ToNNF(c.child).$$
  Если $c.type = AND$, то
  $$ToNNF(node) = ToNNF\big(NOT(c.left) \vee NOT(c.right)\big).$$
  Если $c.type = OR$, то
  $$ToNNF(node) = ToNNF\big(NOT(c.left) \wedge NOT(c.right)\big).$$
  Если $c.type = AX$, то
  $$ToNNF(node) = ToNNF\big(EX\,(NOT(c.child))\big).$$
  Если $c.type = EX$, то
  $$ToNNF(node) = ToNNF\big(AX\,(NOT(c.child))\big).$$
  Если $c.type = \Box_i$ или $\Box_{\leq}$, то
  $$ToNNF(node) = ToNNF\big(\Diamond_i(NOT(c.child))\big)$$
  или
  $$ToNNF(node) = ToNNF\big(\Diamond_{\leq}(NOT(c.child))\big).$$
  Если $c.type = \Diamond_i$ или $\Diamond_{\leq}$, то
  $$ToNNF(node) = ToNNF\big(\Box_i(NOT(c.child))\big)$$
  или
  $$ToNNF(node) = ToNNF\big(\Box_{\leq}(NOT(c.child))\big).$$
  Во всех остальных случаях (атомы) NOT остаётся без изменений.
– **Все прочие узлы, исключая** $\boldsymbol{AU}$ **и** $\boldsymbol{EU}$**.**
  Если у узла есть потомки, то для каждого $d \in node.children$ выполнить
  $$d \leftarrow ToNNF(d).$$
  Затем вернуть исходный узел.

Отметим, операторы AU и EU не подлежат раскрытию, так как подобные действия могут порождать бесконечные циклы, согласно [5].

Определим *модальную степень* $d(\varphi)$ формулы φ как оператор, характеризующий число вложенных операторов вида AX/EX:

1) $d(p) = 0$, $p \in Prop$;
2) $d(\circ\varphi) = d(\varphi)$, где $\circ \in \{\neg, \Box_i, \Box_\leq, \Diamond_i, \Diamond_\leq\}$;
3) $d(\varphi \odot \psi) = max(d(\varphi), d(\psi))$, где $\odot \in \{\wedge, \vee, AU, EU\}$;
4) $d(\bullet\varphi) = d(\varphi) + 1$, где $\bullet \in \{AX, EX\}$.

Определим *степень ветвления* $b(\varphi)$ формулы φ как оператор, отражающий минимальное число ветвлений в подходящей модели для формулы φ:
- $b(p) = 0$, $p \in Prop$;
- $b(\circ\varphi) = b(\varphi)$, где $\circ \in \{\neg, AX, EX, \Box_\leq, \Diamond_\leq\}$;
- $b(\varphi \odot \psi) = max(b(\varphi), b(\psi))$, где $\odot \in \{\wedge, \vee, AU, EU\}$;
- $b(\bullet\varphi) = b(\varphi) + 1$, где $\bullet \in \{\Box_i, \Diamond_i\}$;

Для проверки формулы φ достаточно создать конечную модель, состоящую из D временных слоев, где $D = d(\varphi) + 1$ (с увеличением на единицу за счет корневого состояния) и $B = b(\varphi)$ ветвлений. Если $b(\varphi) = 0$, то $B = 1$ и модель не требует дополнительных агентов (в ней нет ветвлений).

Определение 9. *Минимальной моделью формулы φ называем модель $\mathcal{M}$ с корнем s_0, такую что $\langle \mathcal{M}, s_0 \rangle \Vdash_V \varphi$ и она является минимальной сложностной реализацией φ, т. е. достигает границ по глубине и ветвлению: $D(\mathcal{M}) = d(\varphi) + 1$, $B(\mathcal{M}) = b(\varphi)$, где $D(\mathcal{M})$ — число слоёв по отношению $R_\leq$, а $B(\mathcal{M})$ — число ветвлений в модели (иначе, число i-х агентов). Среди таких моделей выбираем с минимальным числом состояний (в частности, $|W| \leq B \cdot D$).*

Лемма 1. *Для построения минимальной модели формулы φ в логике $\mathcal{CTLK}^{Rel}$ требуется не больше чем $B \cdot D$ состояний.*

Доказательство. Модальные операторы в формуле вида $\Box_i$ и $\Diamond_i$ определяют количество ветвлений модели. Временные операторы вида AX и EX позволяют оценить количество слоёв. Остальные модальные операторы транзитивны и рефлексивны, они не влияют на размерность минимальной модели. Тогда мы можем рассматривать произведение $B \cdot D$ для определения количества состояний. $\Box$

Имеет место следующее

Следствие 1. *Для задачи построения минимальной модели верны следующие эквивалентности:*
- $A(\alpha\, U\, \beta) \equiv (\beta \vee (\alpha \wedge AX\, \beta)$;

– $E(\alpha\, U\, \beta) \equiv (\beta \vee (\alpha \wedge EX\, \beta)$.

Доказательство. Минимальная модель строится «самым коротким» способом. Рассмотрим первую эквивалентность в виде формулы $\varphi := A(\alpha\, U\, \beta)$, тогда обозначим

$$d(\varphi) = max(d(\alpha), d(\beta)) = D \text{ и } b(\varphi) = max(b(\alpha), b(\beta)) = B.$$

Модель удовлетворяет φ, если выполняется β в состоянии s. Нам необходимо обеспечить ровно два возможных варианта:

1) β истинна уже в текущем состоянии s. Тогда условие Until выполняется и дополнительные слои модели создавать не требуется.

2) β ложна в s, но α истинна. В этом случае, чтобы сохранить модель минимальной по глубине, мы обязаны реализовать Until сразу на следующем шаге. Поэтому на всех следующих состояниях должно выполняться β (таких состояний ровно B). Это делает формулу β истинной на глубине D, а глубже состояний нет.

Любая попытка «растянуть» выполнение β дальше приводит к добавлению новых состояний, следовательно, нарушается минимальность модели по произведению $B \cdot D$.

Обратно

1) Если в s истинна β, то условие Until выполняется.

2) Если истинна $\alpha \wedge AX\beta$, то
   - α верно в s;
   - $AX\beta$ гарантирует, что на всех следующих состояниях выполняется β.

   Значит, β встречается на шаге D, а до него выполняется α, поэтому $A(\alpha U \beta)$ истинна в мире s.

Аналогичные рассуждения и для второй эквивалентности. □

Для формулы, содержащей операторы AU и EU, рассмотрим следующий алгоритм:

1) рекурсивно обходим дерево AST;

2) когда встречаем узел типа AU или EU, извлекаем из него формулы p (левая) и q (правая) и приводим их к NNF;

3) строим замену по правилу
   - $A(\alpha\, U\, \beta) \equiv (\beta \vee (\alpha \wedge AX\, \beta)$,
   - $E(\alpha\, U\, \beta) \equiv (\beta \vee (\alpha \wedge EX\, \beta)$;

4) получив новую вершину, снова запускаем алгоритм на ней, чтобы раскрыть вложенные AU, EU;

5) для всех прочих узлов рекурсивно обрабатываем потомков.

В результате в конце в дереве не остается операторов AU и EU.

Итогом действия приведённых выше алгоритмов является формула, преобразованная в NNF, где отрицания содержатся только в конечных узлах нашего дерева.

Далее построим модель $M = \langle F, V \rangle$, исходя из структуры AST глубины D, по следующему алгоритму [10]:

– **Пропозициональные переменные.**
  Если узел — p или $\neg p$, добавить соответствующую метку в $V(s)$.
– **Унарные модальные операторы AX/EX.**
  Если текущий уровень $t = D$, выдать ошибку «максимальная глубина достигнута». Иначе создать новое состояние

$$s' = MakeName(s, t+1),$$

  добавить s' в C_{t+1} и пару (s, s') в $R_{\leq}$, затем рекурсивно вызвать

$$BuildModelFromAST(node.child,\ s',\ t+1).$$

– **Другие модальные операторы.**
  Если для s уже есть требование другого модального оператора или задействован агент, то создать новое состояние

$$s' = MakeName(s)$$

  на том же уровне t, добавить s' в C_t, ввести в R_i связи (s_p, s') и (s', s') (где $s_p \in C_{t-1}$) и рекурсивно вызвать

$$BuildModelFromAST(node.child,\ s',\ t).$$

  В противном случае просто добавить требование из узла в $V(s)$.
– **Бинарные операторы.**
  Для $\wedge$ рекурсивно вызвать $BuildModelFromAST$ для левой и правой подформул в том же (s, t). Для $\vee$ проверить

$$CanBuild(node.left):$$

  если истина, обрабатывать левую ветвь, иначе — правую.

Таким образом, после работы алгоритма мы получаем структуру, являющуюся моделью логики. Осуществим проверку корректности означивания во всех построенных состояниях:

– **Обход состояний.** Для каждого $s \in W$ выполняем следующий шаг.
– **Обход переменных.** Для каждого $p \in AP$ проверяем условие $p \in V(s)\ \wedge\ \neg p \in V(s)$.
  Если оно выполняется, сразу возвращаем

Противоречие в состоянии s: p и $\neg p$ одновременно.

– **Итог.** Если ни в одном состоянии не обнаружено противоречие, возвращаем «модель корректна».

Результат работы алгоритма — ответ о корректности модели.

5.1. Оценка сложности

Таким образом, оценивая сложность всех алгоритмов, получаем:
Построение AST. Дерево разбор формулы имеет размер, пропорциональный числу подформул в замыкании Γ. Количество подформул растёт линейно относительно длины исходной формулы [2], поэтому обход всех подформул для построения AST требует линейного времени:

$$O(|\Gamma|).$$

Преобразование в NNF. Перевод формулы в NNF выполняется за линейное время по размеру формулы, так как алгоритм проходит по формуле один раз, распространяя отрицания только до пропозициональных переменных: $O(|\Gamma|)$.
Раскрытие Until-операторов. Каждая подформула с Until-оператором может быть эквивалентно развернута с лишь линейным увеличением размера, следствие 1. Таким образом, $O(|\Gamma|)$.
Построение модели. Генерация (обход) состояний модели до глубины D при ветвлении B требует времени, пропорционального числу узлов и переходов. Следовательно, временная сложность составляет

$$O(B \cdot D).$$

Проверка корректности модели. Алгоритм проходит по всем состояниям модели и для каждого состояния проверяет маркировку на противоречия. Отметка $B \cdot D$ состояний по одной пропозициональной переменной занимает $O(B \cdot D)$, поэтому если $|AP|$ — число пропозициональных переменных формулы φ, то общая оценка:

$$O(B \cdot D \cdot |AP|).$$

Синтаксические преобразования (построение AST, NNF и раскрытие Until) выполняются за время $O(|\Gamma|)$, а построение и проверка модели — за время $O(B \cdot D)$ и $O(B \cdot D \cdot |AP|)$ соответственно, что является полиномиальной сложностью от параметров B, D и $|\Gamma|$.

6. Заключение

В рамках работы был создан алгоритм построения моделей для реляционной логики $\mathcal{CTLK}^{Rel}$. Показано:

– минимальная модель любой выполнимой формулы содержит не более $B \cdot D$ состояний, где B — степень ветвления формулы, а D — число временных слоёв;

– класс конечных $\mathcal{CTLK}^{Rel}$-фреймов рекурсивно перечислим, а вычислимая граница ($2^{|\Gamma|}$) на их размер даёт разрешимость логики;
– асимптотика всех этапов алгоритма соответствует теоретическим ожиданиям и выражается через параметры $|\Gamma|, B, D$.

Тем самым доказана разрешимость $\mathcal{CTLK}^{Rel}$. Полученные результаты делают $\mathcal{CTLK}^{Rel}$ удобным инструментом практического model-checking и анализа многоагентных распределённых систем, а также позволяют рассчитывать на реальное использование логики $\mathcal{CTLK}^{Rel}$ в верификации протоколов распределённых вычислений и систем с временными ограничениями.

Список источников

1. Башмаков С. И., Смелых К. А. Реляционная версия многоагентной логики деревьев вычислений CTLK // Известия Иркутского государственного университета. Серия Математика. 2024. Т. 47. С. 78–92. https://doi.org/10.26516/1997-7670.2024.47.78
2. Baier C., Katoen J. P. MIT Press, 2008.
3. Blackburn P., De Rijke M., Venema Y. Modal logic. Cambridge Univ. Press, 2001. Vol. 53. 554 p.
4. Logic-based technologies for multi-agent systems: a systematic literature review / R. Calegari [et al.] // Autonomous Agents and Multi-Agent Systems. 2021. Vol. 35, N 1. P. 1.
5. Chagrov A., Zacharyaschev M. Modal Logic. Oxford Univ. Press. 1997, 605 p.
6. Darwiche A. A logical approach to factoring belief networks. // Proceedings of the 8th Internat. Conference on Principles of Knowledge Representation and Reasoning. 2002. Vol. 2. P. 409–420.
7. Dima C. Revisiting satisfiability and model-checking for CTLK with synchrony and perfect recall // Computational Logic in Multi-Agent Systems. CLIMA 2008. Lecture Notes in Computer Science. 2008. Vol. 5405. P. 117–131. https://doi.org/10.1007/978-3-642-02734-5_8
8. Emerson E. A., Halpern J. Y. Decision procedures and expressiveness in the temporal logic of branching time. // Proceedings of the 14th Annual ACM Symposium on Theory of Computing. 1982. P. 169–180.
9. He L., Liu G., Zhou M. Petri-net-based model checking for privacy-critical multiagent systems // Transactions on Computational Social Systems. 2022. Vol. 10, N 2. P. 563–576.
10. Limón Y., Bárcenas E., Benítez-Guerrero E. Depth-first reasoning on trees // Computación y Sistemas. 2018. Vol. 22. N 1. P. 189–201.
11. Srinivasan M., Coogan S., Egerstedt M. Control of multi-agent systems with finite time control barrier certificates and temporal logic // 2018 IEEE Conference on Decision and Control (CDC). IEEE, 2018. C. 1991–1996.

References

1. Bashmakov S.I., Smelykh K. A Relational version of the multi-agent computation tree logic CTLK. *The Bulletin of Irkutsk State University. Series Mathematics*, 2024, vol. 47, pp. 78–92. https://doi.org/10.26516/1997-7670.2024.47.78

2. Baier C., Katoen J.P. *MIT press*, 2008.
3. Blackburn P., De Rijke M., Venema Y. Modal logic. *Cambridge Univ. Press*, 2001, vol. 53, 554 p.
4. Calegari R. et al. Logic-based technologies for multi-agent systems: a systematic literature review *Autonomous Agents and Multi-Agent Systems*, 2021, vol. 35, no. 1, p. 1.
5. Chagrov A., Zacharyaschev M. *Modal Logic*, Oxford Univ. Press, 1997, 605 p.
6. Darwiche A. A logical approach to factoring belief networks. *Proceedings of the 8ts Internat. Conference on Principles of Knowledge Representation and Reasoning*, 2002, vol. 2, pp. 409–420.
7. Dima C. Revisiting satisfiability and model-checking for CTLK with synchrony and perfect recall. *Computational Logic in Multi-Agent Systems. CLIMA 2008. Lecture Notes in Computer Science*, 2008, vol. 5405, pp. 117–131. https://doi.org/10.1007/978-3-642-02734-5_8
8. Emerson E.A., Halpern J.Y. Decision procedures and expressiveness in the temporal logic of branching time. *Proceedings of 14th annual ACM symposium on Theory of computing*, 1982, pp. 169–180.
9. He L., Liu G., Zhou M. Petri-net-based model checking for privacy-critical multiagent systems. *Transactions on Computat. Social Systems*, 2022, vol. 10, no. 2, pp. 563–576.
10. Limón Y., Bárcenas E., Benítez-Guerrero E Depth-first reasoning on trees. *Computación y Sistemas*, 2018, vol. 22, no. 1, pp. 189–201.
11. Srinivasan M., Coogan S., Egerstedt M. Control of multi-agent systems with finite time control barrier certificates and temporal logic *2018 IEEE Conference on Decision and Control (CDC)*. IEEE, 2018, pp. 1991–1996.

Об авторах

Башмаков Степан Игоревич,
канд. физ.-мат. наук, доц.,
Сибирский федеральный
университет, Красноярск, 660041,
Российская Федерация,
krauder@mail.ru,
https://orcid.org/orcid.org/0000-0002-3354-0383

Смелых Кирилл Александрович,
студент, Сибирский федеральный
университет, Красноярск, 660041,
Российская Федерация,
lastth@yandex.ru,
https://orcid.org/0009-0007-3228-2806

About the authors

Stepan I. Bashmakov, Cand. Sci.
(Phys.-Math.), Assoc. Prof.,
Siberian Federal University,
Krasnoyarsk, 660041,
Russian Federation,
krauder@mail.ru,
https://orcid.org/orcid.org/0000-0002-3354-0383

Kirill A. Smelykh,
Student, Siberian Federal University,
Krasnoyarsk, 660041,
Russian Federation,
lastth@yandex.ru,
https://orcid.org/0009-0007-3228-2806

Поступила в редакцию / Received 10.07.2025
Поступила после рецензирования / Revised 10.10.2025
Принята к публикации / Accepted 24.10.2025