Computing
https://doi.org/10.1007/s00607-024-01258-1

REGULAR PAPER

Solving the SAT problem with the string multiset rewriting calculus

Péter Battyányi[1]

Received: 17 April 2023 / Accepted: 7 January 2024

Abstract
In this paper, we develop computing machinery within the framework of the String Multiset Rewriting calculus (SMSR), as defined by Barbuti et al. [4], to solve the SAT problem in linear time regarding the number of variables of a given conjunctive normal form. This shows that SMSR can be considered a computational model capable of significantly reducing the time requirement of classical decision problems.

Keywords String multiset rewriting · Biologically motivated computing · SAT-problem

Mathematics Subject Classification 68Q07 · 68Q42 · 68Q85

1 Introduction

In the past few years, many computational models have been developed to model biological processes at the systems level, which involves describing the elements of complex systems and their mutual behaviour and interactions. Traditional approaches rely on mathematical methods mainly based on setting up ordinary differential equations in connection with the chemical reactions. However, these systems can encounter difficulties when it comes to finding feasible models or establishing executable simulations. Computer science offers various formalisms based on high-level description of biological phenomena. For instance, the κ-calculus of Danos and Laneve [9] serves as an example of a formal language that models protein interaction. On the other hand, the biochemical stochastic π-calculus [18] allows for both qualitative and quantitative investigations of composite systems. Additionally, the brane calculi defined by Cardelli [7] or the membrane systems invented by Gh. Păun [17] are excellent models for describing cell-level processes. The calculus

✉ Péter Battyányi
battyanyi.peter@inf.unideb.hu

[1] Department of Computer Science, Faculty of Informatics, University of Debrecen, Debrecen, Hungary

Published online: 06 February 2024

of looping sequences [5] falls into this category: these models allow the extensive use of mathematical tools in describing biological processes. However, constructing efficient simulators for these high level languages, such as the calculus of looping sequences and the brane calculi, can be quite challenging.

To meet the requirement of finding a formalism that is both expressive enough to describe biological processes and maintains language simplicity, some authors choose to base their calculi on multisets as objects. Barbuti et al. [4, 5] developed formalisms centered around pattern matching as the main tool underlying the evolvement of multisets. Others attempt to integrate rule-based modeling with a chemical reaction-type formalism [20] or enhance the labeled transition system with a regulating mechanism for rules [21]. The calculus developed in [4] was called the String Multiset Rewriting calculus (SMSR). As part of a series of investigation conducted by the author regarding biologically motivated calculi [3], especially, the string multiset rewriting formalism, this paper also adopts this calculus. Interestingly, independent of our current account, the investigation initiated in [22], aiming to merge regulated rules with rewrite mechanisms based on pattern matching, also leverages the formalism introduced by Barbuti et al. The SMSR calculus arouse as a simplified version of the calculus of looping sequences [5]. Due to its simplicity and the powerful pattern matching mechanism, SMSR can serve as the foundation for a declarative description of biological processes and thus provide an elegant solution for establishing the background formalism for a mathematically precise treatment. Hopefully, its overall construction may prove to be simple enough to facilitate the development of efficient future simulators. Specifically, we work with multisets of string elements that constitute the underlying objects of the calculus. The central tool for manipulating multisets is the maximal matching operator, which allows for the replacement of complete submultisets where the strings share the same prefix. The idea is that, by capturing the tree-like structure of terms, a string can represent a path from the root to one of the leaves, where at the leaves the elements constituting the terms can be found. If we simultaneously replace all the strings with a common prefix with a multiset of strings starting with the same prefix, then the tree-like structure is preserved. This mechanism corresponds to rewriting processes in term rewriting systems. In addition to the simplicity of the elements of the calculus, as we have mentioned before, another compelling reason for the SMSR calculus to be considered a promising candidate for potential future implementations in higher-level languages is the inclusion of pattern matching within the maximal matching operator. In their paper, Barbuti et al [4] showed how the calculus of looping sequences formalism can be corresponded to string multiset rewriting. Additionally, Barbuti et al. pointed out in [5] that their method can serve as a template for translating other formalisms, like P systems or brane calculi, into the framework of string multiset rewriting. Therefore, even though the SMSR calculus is based on a simple structure with just one rewrite operator, it still has the capability to express the interactions between elements that more complex formalisms are capable of.

There have been numerous results based on parallel and distributed computational models, in which the SAT problem has been shown to be solvable in fewer than an exponential number of steps relative to the input size, considering only the computational steps allowed by the underlying model [6, 11–13, 16, 23]. Most of

them rely on a quite complex formalism constructed in a biologically inspired computational model. This paper demonstrates that even a simple formalism like the string multiset rewriting calculus can achieve a level of parallelization sufficient for effectively solving a hard problem like SAT. It does so by establishing the existence of a multiset such that, for any conjunctive normal form (CNF) C with n variables and m conjuncts, the constructed multiset comprises $3n + 1$ submultisets, each consisting of at most m strings. Additionally, we can provide $n + 1$ rewrite rules such that, by applying these rules to the initial multiset, we can determine in $n + 1$ rewriting steps whether C is satisfiable. The construction is uniform: the rules established depend only on the values of n and m. Furthermore, the multiset assigned to the CNF can be constructed in polynomial time with respect to n and m. However, the size of the rules is exponential in n, which implies that an exponential space is needed for an implementation. In addition, the supposition that each rule application takes constant time is crucial for maintaining the efficiency. Taking all of this into consideration, SMSR is a truly parallel computational model. With the aforementioned assumptions, it is capable of significantly reducing computational time, provided we accept the exponential time hypothesis, which posits that *SAT* cannot be solved in subexponential time. Naturally, this computational model of string rewriting involves operations that are hypothetical considering our current computational capabilities and are very costly on sequential computational architectures. The majority of the work is accomplished by the highly parallel rules of the multiset rewriting calculus, which operate using pattern matching and the evaluation of patterns to obtain concrete multisets as values.

2 String multiset rewriting

In this section, we will provide a brief introduction to the calculus of String MultiSet Rewriting (SMSR), following the presentation in [4]. Initially, we provide some of the relevant definitions concerning multisets.

2.1 Multisets

In this subsection, we briefly recall the main definitions regarding multisets over a finite nonempty set. Let $\mathbb{N}$ denote the set of natural numbers and $\mathbb{N}^+$ denote the set of positive integers, respectively, and let U be a finite nonempty set. A multiset M over U is a pair $M = (U,f)$, where the mapping $f : U \to \mathbb{N}$ gives the multiplicity of each element $a \in U$. If $f(a) = 0$ for each $a \in U$, then M is the empty multiset. The number of elements in a multiset M, that is, the elements $a \in U$ with positive multiplicity is denoted by $|M|$ and we refer to this number as the cardinality of M. In our case, the cardinality of a string multiset M will be the number of strings in M.

Next, we define some elementary operations on multisets. Let $M_1 = (U,f_1), M_2 = (U,f_2)$. Then $(M_1 \sqcap M_2) = (O,f)$ where $f(a) = \min\{f_1(a),f_2(a)\}$; $(M_1 \sqcup M_2) = (O,f')$, where $f'(a) = \max\{f_1(a),f_2(a)\}$; $(M_1 \oplus M_2) = (O,f'')$, where $f''(a) = f_1(a) + f_2(a)$; $(M_1 \ominus M_2) = (O,f''')$ where $f'''(a) = \max\{f_1(a) - f_2(a), 0\}$;

and $M_1 \sqsubseteq M_2$, if $f_1(a) \leq f_2(a)$ for all $a \in O$. For our purposes, we will modify the subtraction operator in the following way: when $M_1 \sqsubseteq M_2$, then the result of the subtraction should be a designated element $\Lambda \in U$, instead of 0. We will denote the modified subtraction operator as $\ominus_\Lambda$. For example, let $U = \{a, b, c, \Lambda\}$ be the underlying set and suppose $M_1 = a^3bc^2$ and $M_2 = a^3b^2c$. Then $M_1 \ominus_\Lambda M_2 = c$. On the other hand, if $M_1' = a^3bc^2$ and $M_2' = a^3b^2c^2$, then $M_1' \ominus_\Lambda M_2' = \Lambda$.

2.2 Fundamental notions

In their paper [4], Barbuti et al. introduced the notion of the String MultiSet Rewriting calculus (SMSR). Initially, we establish the syntax, followed by the development of a theory based on multiset rewriting, subject to certain structural congruence relations. Naturally, the multisets can evolve according to the rewriting rules specified in the next subsection, modulo these congruence relations. The resulting computational model is commutative and associative with respect to the multiset union operator. In the definition below, we assume there is an underlying alphabet $\mathcal{E}$, which can be either finite or infinite.

Definition 1 Multisets M and strings S over an alphabet $\mathcal{E}$ are defined by the following grammar:

$$\begin{array}{rcl} M & = & S \mid (M \mid M) \\ S & = & \varepsilon \mid e \mid S \cdot S \end{array}$$

where ε represents the empty string, e stands for a generic element of the alphabet $\mathcal{E}$, $\cdot$ denotes string concatenation, and $\mid$ is the multiset union. The set comprising all multisets is denoted by $\mathbf{M}$, while the set of all strings is denoted by $\mathbf{S}$. An arbitrary element of $\mathbf{S}$ is represented by lowercase Greek letters such as ξ, η, and so on. We may omit the operation $\cdot$ between the elements of a string if there is no potential for confusion.

The multiset operation $\mid$ functions as the standard multiset constructor. We assume that $\mid$ is both commutative and associative, allowing us to omit parentheses when no confusion arises. The following structural congruence relation encapsulates the associativity of $\cdot$ and $\mid$, the commutativity of $\mid$, and the neutral role of ε as both an element and a string.

Definition 2 Let ξ, ξ_1, ξ_2, ξ_3 be strings and M, M_1, M_2, M_3 be multisets. The structural congruence relation $\equiv$ is the least congruence relation on multisets with the following properties:

$$\xi_1 \cdot (\xi_2 \cdot \xi_3) \equiv (\xi_1 \cdot \xi_2) \cdot \xi_3 \qquad \xi \cdot \varepsilon \equiv \xi$$
$$M_1 \mid M_2 \equiv M_2 \mid M_1 \qquad M_1 \mid (M_2 \mid M_3) \equiv (M_1 \mid M_2) \mid M_3 \qquad M \mid \varepsilon \equiv M$$

Keeping all of this in mind, we can now introduce the concept of patterns, which will be the key notion in the definition of the rewriting rules. We will

assume an infinite supply of variables denoted as $\mathcal{V} = \mathcal{V}_{\mathcal{E}} \cup \mathcal{V}_{\mathbf{S}} \cup \mathcal{V}_{\mathbf{M}}$, where $\mathcal{V}_{\mathcal{E}} = x, y, z, \ldots$ represents the infinite set of *element variables*, $\mathcal{V}_{\mathbf{S}} = \widetilde{x}, \widetilde{y}, \widetilde{z}, \ldots$ stands for the infinite set of *string variables*, and $\mathcal{V}_{\mathbf{M}} = X, Y, Z, \ldots$ signifies the infinite set of *multiset variables*. Multiset patterns are defined as follows.

Definition 3 We define multiset patterns MP and string patterns SP over an alphabet $\mathcal{E}$ by the grammar below:

$$\begin{array}{rcl} MP & = & SP \mid MP \mid MP \mid \{|SP|\}_X \\ SP & = & \varepsilon \mid e \mid SP \cdot SP \mid \widetilde{x} \mid x \end{array}$$

Patterns are the general instructions for evaluating multiset terms. A rewrite rule consists of a pair of patterns, where the first pattern matches the term to be modified and the second one determines the result of the rule application.

Given an evaluation $\sigma : \mathcal{V}_{\mathcal{E}} \cup \mathcal{V}_{\mathbf{S}} \to \mathbf{S}$ together with a function $\rho : \mathcal{V}_{\mathbf{M}} \to \mathbb{N}$, we can lend meaning to the multiset patterns as follows. The pattern expansion function $\langle \_ \rangle_{\rho}^{\sigma}$ assigns a multiset over $\mathcal{E}$ to the multiset pattern. Multiset patterns of the form $\{|SP|\}_X$ correspond to a union of a sequence of patterns, all having the same SP prefix followed by the evaluation of different string variables.

In the following definition, we provide the interpretation of multiset patterns in relation to a function $\rho : \mathcal{V}_{\mathbf{M}} \to \mathbb{N}$ and an evaluation $\sigma : \mathcal{V}_{\mathcal{E}} \cup \mathcal{V}_{\mathbf{S}} \to S$. Intuitively, the function σ ensures that all the string and element variables appearing in a pattern are instantiated, while the function ρ determines the number of strings in the multiset obtained as the result of evaluating a pattern of the form $\{|S|\}_X$.

Definition 4 Assume a function $\rho : \mathcal{V}_{\mathbf{M}} \to \mathbb{N}$ and an evaluation $\sigma : \mathcal{V}_{\mathcal{E}} \cup \mathcal{V}_{\mathbf{S}} \to \mathbf{S}$ are given. The pattern expansion function $\langle \_ \rangle_{\_}^{-} : MP \times (\mathcal{V}_{\mathbf{M}} \to \mathbb{N}) \times (\mathcal{V}_{\mathcal{E}} \cup \mathcal{V}_{\mathbf{S}} \to \mathbf{S}) \to \mathbf{M}$ is recursively defined as follows:

(1) $\langle SP \rangle_{\rho}^{\sigma} = \sigma(SP)$
(2) $\langle \{|SP|\}_X^{\sigma} \rangle_{\rho} = \sigma(SP) \cdot \sigma(\widetilde{x}_1) \mid \ldots \mid \sigma(SP) \cdot \sigma(\widetilde{x}_{\rho(X)})$ where $\widetilde{x} \in \mathcal{V}_{\mathbf{S}}$
(3) $\langle MP_1 \mid MP_2 \rangle_{\rho}^{\sigma} = \langle MP_1 \rangle_{\rho}^{\sigma} \mid \langle MP_2 \rangle_{\rho}^{\sigma}$.

In the previous definition, if a multiset variable X is given, then $\rho : \mathcal{V}_{\mathbf{M}} \to \mathbb{N}$ selects new string variables $\widetilde{x}_1, \ldots, \widetilde{x}_{\rho(X)}$ for the expansion. To be more precise, $\rho : \mathcal{V}_{\mathbf{M}} \to (\mathcal{V}_{\mathbf{S}})^n$ holds for some n determined by ρ. Moreover, we assume that, if X and Y are distinct multiset variables, then $\rho(X)$ and $\rho(Y)$ use different string variables, that is, if $\rho(X) = n$ and $\rho(Y) = m$, then the variable sets $\{\widetilde{x}_1, \ldots, \widetilde{x}_n\}$ and $\{\widetilde{y}_1, \ldots, \widetilde{y}_m\}$ appearing in the expansion of $\rho(X)$ and $\rho(Y)$ are disjoint. While we employ the informal notation introduced in the definition, we bear in mind the stipulations just agreed upon. We will now illustrate the previous definition by an example.

Example 5 Let $\mathcal{E} = \{a, b, c\}$ be the element set. Then $MP_1 = x \cdot y \mid a \cdot x$ and $MP_2 = \{|a|\}_X \mid \widetilde{x}$ are examples for multiset patterns. If we set $\sigma(x) = \sigma(y) = c$,

$\sigma(\widetilde{x}) = a \cdot b$, $\rho(X) = 2$ and $\sigma(\widetilde{x}_1) = a$ and $\sigma(\widetilde{x}_2) = a \cdot a$, then we are able to determine the values of the patterns MP_1 and MP_2 with respect to the values of the functions ρ and ρ. Namely, $\langle MP_1 \rangle_\rho^\sigma = c \cdot c \mid a \cdot c$ and $\langle MP_2 \rangle_\rho^\sigma = a \cdot a \mid a \cdot a \cdot a \mid a \cdot b$.

2.3 Rewrite rules

In the following subsection, we will define the rewrite rules. To ensure that this paper is self-contained, we provide a precise definition of rewriting. Intuitively, the result of applying a rewrite rule involves substituting a multiset M with another multiset M'. This process is governed by a rule $(MP_1, MP_2) \in \mathfrak{R}$, where MP_1 and MP_2 are multiset patterns such that $\langle MP_1 \rangle_\rho^\sigma = M$ and $\langle MP_2 \rangle_\rho^\sigma = M'$ hold, with some ρ and σ. To enhance understanding, we introduce several technical definitions that are based on the paper [4].

Definition 6

1. Let $Var(MP)$ denote the set of variables appearing in a multiset pattern with the provision that $Var(\{|SP|\}_X) = Var(SP) \cup \{\widetilde{x}_i \mid i \in \mathbb{N}\}$. For example, $Var(a \cdot \widetilde{x} \mid y \mid \{|\varepsilon|\}_Y) = \{\widetilde{x}, y\} \cup \{\widetilde{y}_i \mid i \in \mathbb{N}\}$.
2. Let $Symbols(M)$ denote the set of elements of $\mathcal{E}$ that appear in the multiset pattern or multiset M. For instance, $Symbols(a \cdot x | a \cdot b | \{|d|\}_X) = \{a, b, d\}$. We assume that $Symbols$ extends to a set of multiset patterns or multisets and we define the set of fresh names for a multiset M as $\mathcal{E} \setminus Symbols(M)$.
3. A rewrite rule is a pair (MP, MP') of patterns such that MP is non-empty. An instantiation is a function $\sigma : \mathcal{V}_\mathcal{E} \cup \mathcal{V}_\mathbf{S} \to \mathbf{M}$.
4. A multiset pattern MP is ground if and only if $Var(MP) = \emptyset$. A rule $R = (MP, MP')$ is ground if and only if both MP and MP' are ground. We write $Var(R) = Var(MP) \cup Var(MP')$ and $Symbols(R) = Symbols(MP) \cup Symbols(MP')$.
5. Let $R = (MP, MP')$. Then $FV(R) = \{v \mid v \in Var(MP') \setminus Var(MP)\}$ and $BV(R) = Var(R) \setminus FV(R)$.

Before delving into the semantics of the rewrite step, we aim to clarify the related concepts in a more intuitive manner.

Definition 7 The multisets M and M' can be matched by applying the rule (MP_1, MP_2) if $\langle MP_1 \rangle_\rho^\sigma = M$ and $\langle MP_2 \rangle_\rho^\sigma = M'$, where $\rho : \mathcal{V}_\mathbf{M} \to \mathbb{N}$ is an auxiliary function and σ is an instantiation. Let us assume $M \sqsubseteq N$, that is, N is a containing multiset, and M matches M'. When this is the case, we obtain the result of applying the rewrite rule by replacing the submultiset M of N by M'. Furthermore, we make sure that no string in N that does not belong to M begins with a prefix shared by all the strings in M.

Now, we are ready to define the semantics of the rewrite rules.

Definition 8 Let $R = (MP_1, MP_2) \in \mathfrak{R}$. An application of R can be described as follows. Assume M, M' are multisets such that $\langle MP_1 \rangle_\rho^\sigma = M$ and $\langle MP_2 \rangle_\rho^\sigma = M'$ for some ρ and σ.

1. Assume $Symbols(\sigma(BV(R)) \cup Symbols(R)) \cap Symbols(\sigma(FV(R))) = \emptyset$. Then we write

$$M \xrightarrow{(\{SP\sigma \mid \langle \{|SP|\}_X \rangle_\rho^\sigma \sqsubseteq M\}, Symbols(\sigma(FV(R))))} M'. \quad (1)$$

2. If $M \xrightarrow{(\xi,\zeta)} M'$ and $(\forall S \in \xi)(\nexists S' \in \mathbf{S})(S \cdot S' \sqsubseteq M'')$ and $Symbols(M'') \cap \zeta = \emptyset$, then

$$M'' \mid M \xrightarrow{(\xi,\zeta)} M'' \mid M'. \quad (2)$$

The first item expresses that each substituted occurrence of a maximal matching operator counts in the reduction step. We lead a book-keeping containing the occurrences of the maximal matching operator together with the names occurring as occupied names in the multiset patterns of the rule applied. The second item ensures that, when a string with prefix $SP\sigma$ appears in M, then all strings with the same prefix is treated in the current rewrite step.

Let us illustrate with an example how the rewrite process takes place.

Example 9 Let $M = b \mid b \cdot c \mid b \cdot d$. Let $R_1 = \{|b|\}_X \hookrightarrow a$ and $R_2 = b \cdot \widetilde{x} \mid \{|b|\}_X \hookrightarrow a$ be two rules. Then

1. Then $M \hookrightarrow M'' = a$ is a correct derivation by applying R_1 with $\rho_2(X) = 3$ and $\sigma_2(\widetilde{x_1}) = \epsilon$, $\sigma_2(\widetilde{x_2}) = c$ and $\sigma_2(\widetilde{x_3}) = d$.
2. Likewise, $b \mid b \cdot c \mid b \cdot d \hookrightarrow_{R_2} b \mid a$ with $\sigma_1(\widetilde{x}) = \epsilon$, $\rho_1(X) = 2$ and $\sigma_1(\widetilde{x_1}) = c$, $\sigma_1(\widetilde{x_2}) = d$ is a correct application of R_2 to M.
3. However,

$$b \mid b \cdot c \mid b \cdot d \hookrightarrow_{R_1} a \mid b \cdot d$$

would be an incorrect derivation, since it violates the stipulation that $\langle \{|b|\}_X \rangle_\rho^\sigma$ should be the maximal subterm of M the strings of which start with the common prefix b.

3 Solving the SAT-problem

In this section, we present a decision procedure for the SAT problem that is linear in the number of the variables of the given conjunctive normal form (CNF), taking into account the computational steps performed in the SMSR-calculus. Firstly, we provide a short account of the SAT problem. A formula of the form

$$C = D_1 \wedge D_2 \wedge \cdots \wedge D_m \quad (1 \leq m),$$

is considered a conjunctive normal form if $D_i = A_{i1} \vee A_{i2} \vee \cdots \vee A_{ik_i}$, where A_{ij} is either a propositional variable or its negation ($1 \leq i \leq m, 1 \leq j \leq k_i$) and $k_i \in \mathbb{N}$. We denote variables with lower case letters $x_1, x_2, \ldots$. The SAT problem involves finding at least one assignment, a function assigning truth values to the variables, which satisfies the underlying conjunctive normal form. Since there is a finite number of variables in each CNF, the SAT problem is in general decidable. The challenge lies in finding a method capable of deciding the satisfiability problem for any CNF formula in less than exponential time, at least in principle. Below, we present one based on the SMSR calculus.

Theorem 10 *The SAT problem can be decided in the SMSR calculus in time that is linear in the number of variables.*

Proof Let $C = D_1 \wedge D_2 \wedge \cdots \wedge D_m$ be a conjunctive normal form. Assume there are n distinct variables $\{x_1, \ldots, x_n\}$ appearing in C. We will construct a multiset and the corresponding rewrite rules such that, by starting from this multiset and applying the rules, we can decide in $n + 1$ rule applications whether C is satisfiable. To facilitate comprehension, we will provide a running example throughout the proof.

We will begin by considering the general case. Let $C = D_1 \wedge D_2 \wedge \cdots \wedge D_m$ as defined above. We establish the multiset M_C corresponding to C. Our main idea is to devise a multiset pattern for capturing the interpretations that satisfy C, preserving only those in the end that do not make C true. We keep track of the appearances of x_j ($\neg x_j$, resp.) in the conjuncts $D_1, \ldots, D_m$ with the help of the strings starting with prefix β_j ($\neg\beta_j$, resp.) in M_C. The rules will allow us to develop the multiset M_C through n rule applications. The multiset emerging after the jth rule application leaves us with a guidance on how to choose the values of the variables $x_1, \ldots, x_j$ for an interpretation satisfying all of the conjuncts, if any exist. Taking all of this into consideration, we begin by defining the language and the rewrite rules. The underlying set for the string multisets is $\mathcal{E} = V \cup T \cup A$, where $V = \{\#x_1, \ldots, \#x_n\}$ for the variables in C, and $T = \{t_1, \ldots, t_m\}$ for each of the disjunctions $D_1 \ldots, D_m$ and, in addition, a finite number of auxiliary elements $A = \{\alpha, \lambda\} \cup \{\alpha^i_j \mid 1 \leq i \leq n, 1 \leq j \leq 2^i\} \cup \{\beta_i, \neg\beta_i \mid 1 \leq i \leq m\}$ should be present for the evaluation of the variables in C. Additionally, let $\{u^i_j, v^i_j \mid 1 \leq j \leq m, 1 \leq i \leq n\}$ be element variables. If $\beta \in \mathcal{E}$, then we denote the multiset $\beta \cdot \xi_1 \mid \ldots \mid \beta \cdot \xi_p$ by $\beta \cdot (\xi_1 \mid \ldots \mid \xi_p)$. In what follows, we will omit the outermost parentheses of the multisets if no confusion arises. Moreover, let us write $\Psi_i = u^i_1 \mid \ldots \mid u^i_m$ and $\neg\Psi_i = v^i_1 \mid \ldots \mid v^i_m$, where $1 \leq i \leq n$ and $1 \leq j \leq m$. Let us define the following multisets. Let $1 \leq i \leq n$ be arbitrary, and let us denote by Φ_i the multiset $\Phi_i = \varphi^i_1 \mid \ldots \mid \varphi^i_m$ such that φ^i_j is t_j if $x_i \in D_j$ for some $1 \leq j \leq m$ and undefined otherwise. Similarly, we write $\neg\Phi_i = \neg\varphi^i_1 \mid \ldots \mid \neg\varphi^i_m$, where $\neg\varphi^i_j$ are supposed to be the following elements of $\mathcal{E}$: let $\neg\varphi^i_j$ be t_j if $\neg x_i \in D_j$ and undefined otherwise ($1 \leq j \leq m$). Below, let $\Upsilon = t_1 \mid \ldots \mid t_m$, and, for the sake of simplicity, we write x_i instead of $\#x_i$ ($1 \leq i \leq n$). We now define the initial multiset assigned to C as $M_C = \alpha \mid x_1 \mid \ldots \mid x_n \mid \beta_1 \cdot \Phi_1 \mid \ldots \mid \beta_n \cdot \Phi_n \mid \neg\beta_1 \cdot \neg\Phi_1 \mid \ldots \mid \neg\beta_n \cdot \neg\Phi_n$.

Let us consider the following example. Suppose we have $F = (x_1 \vee x_2) \wedge (x_1 \vee \neg x_2) \wedge (\neg x_1 \vee \neg x_2)$. Then F contains two variables and consists of three elementary disjunctions. In this case, $n = 2$ and $m = 3$. Let D_1, D_2 and D_3 stand for the disjunctions in the given order. The multisets Ψ and $\neg\Psi$ are exclusively determined by the values of n and m: we have $\Psi_i = u_1^i \mid u_2^i \mid u_3^i$ and $\neg\Psi_i = v_1^i \mid v_2^i \mid v_3^i$ for $i \in \{1, 2\}$. Let us now establish the values of Φ_i and $\neg\Phi_i$, where $i \in \{1, 2\}$. Since x_1 appears in D_1 and D_2, we have $\Phi_1 = t_1 \mid t_2$. Furthermore, $\neg x_1$ appears only in D_3, hence $\neg\Phi_1 = t_3$. Similarly, $\Phi_2 = t_1$ and $\neg\Phi_2 = t_2 \mid t_3$. We obtain $M_F = \alpha \mid x_1 \mid x_2 \mid \beta_1 \cdot (t_1 \mid t_2) \mid \beta_2 \cdot t_1 \mid \neg\beta_1 \cdot t_3 \mid \neg\beta_2 \cdot (t_2 \mid t_3)$ for the multiset corresponding to F.

Continuing the discussion of the general case, we now define the rewriting rules. Let the first rule be

$$\begin{aligned}
&\alpha \mid x_1 \mid \beta_1 \cdot \Psi_1 \mid \ldots \mid \beta_n \cdot \Psi_n \mid \neg\beta_1 \cdot \neg\Psi_1 \mid \ldots \mid \neg\beta_n \cdot \neg\Psi_n \rightarrow \\
&\alpha_1^1 \cdot \Upsilon \ominus_\Lambda \alpha_1^1 \cdot \Psi_1 \mid \alpha_2^1 \cdot \Upsilon \ominus_\Lambda \alpha_2^1 \cdot \neg\Psi_1 \mid \\
&\beta_1 \cdot \Psi_1 \mid \ldots \mid \beta_n \cdot \Psi_n \mid \neg\beta_1 \cdot \neg\Psi_1 \mid \ldots \mid \neg\beta_n \cdot \neg\Psi_n.
\end{aligned}$$

Observe that Ψ_i unifies with Φ_i and $\neg\Psi_i$ unifies with $\neg\Phi_i$ $(1 \leq i \leq n)$. Let $1 \leq j \leq m$ be given. Then $\alpha_1^1 \cdot \phi_j^1 \in \alpha_1^1 \cdot \Upsilon \ominus_\Lambda \alpha_1^1 \cdot \Phi_1$ conveys a "negative" information. That is, setting x_1 to true is not enough to guarantee that D_j is satisfied in this case. Similarly, $\alpha_2^1 \cdot \neg\phi_j^1 \in \alpha_1^1 \cdot \Upsilon \ominus_\Lambda \alpha_1^1 \cdot \neg\Phi_1$ if and only if $\neg x_1$ does not appear in D_j, hence, setting x_1 to false will not make D_j true. In general, we define the $i+1$-st rule as follows.

$$\begin{aligned}
&\{|\alpha_1^i|\}_{X_1^i} \mid \{|\alpha_2^i|\}_{X_2^i} \mid \ldots \mid \{|\alpha_{2^i}^i|\}_{X_{2^i}^i} \mid x_{i+1} \\
&\mid \beta_1 \cdot \Psi_1 \mid \ldots \mid \beta_n \cdot \Psi_n \mid \neg\beta_1 \cdot \neg\Psi_1 \mid \ldots \mid \neg\beta_n \cdot \neg\Psi_n \rightarrow \\
&\{|\alpha_1^{i+1}|\}_{X_1^i} \ominus_\Lambda \alpha_1^{i+1} \cdot \Psi_{i+1} \mid \\
&\{|\alpha_2^{i+1}|\}_{X_1^i} \ominus_\Lambda \alpha_2^{i+1} \cdot \neg\Psi_{i+1} \mid \\
&\ldots \\
&\{|\alpha_{2^{i+1}-1}^{i+1}|\}_{X_{2^i}^i} \ominus_\Lambda \alpha_{2^{i+1}-1}^{i+1} \cdot \Psi_{i+1} \mid \\
&\{|\alpha_{2^{i+1}}^{i+1}|\}_{X_{2^i}^i} \ominus_\Lambda \alpha_{2^{i+1}}^{i+1} \cdot \neg\Psi_{i+1} \mid \\
&\beta_1 \cdot \Psi_1 \mid \ldots \mid \beta_n \cdot \Psi_n \mid \neg\beta_1 \cdot \neg\Psi_1 \mid \ldots \mid \neg\beta_n \cdot \neg\Psi_n,
\end{aligned}$$

where the right hand side of the rule is obtained from the left hand side in the following way. Let $\{|\alpha_j^i|\}_{X_j^i}$ be a member of the left hand side. We then add the following multiset patterns to the right hand side of the rule. We form the multiset patterns $\{|\alpha_{2j-1}^{i+1}|\}_{X_j^i}$ and $\{|\alpha_{2j}^{i+1}|\}_{X_j^i}$, respectively. Afterward, we conduct subtraction using the multisets $\alpha_{2j-1}^{i+1} \cdot \Psi_{i+1}$ and $\alpha_{2j}^{i+1} \cdot \neg\Psi_{i+1}$. This process entails that, when we instantiate the variables of the multiset patterns through the actual sequence of rule applications, a multiset pattern of the form $\{|\alpha_j^i|\}_{X_j^i}$ transforms into a multiset of strings beginning with α_j^i. more precisely, it becomes $\alpha_j^i \cdot (\theta_1 \mid \ldots \mid \theta_l)$ for some $\Theta_j^i = \theta_1 \mid \ldots \mid \theta_l$.

In the new multiset, we simply replace α_j^i at the beginning of the string with α_{2j-1}^{i+1} and, in the next step, with α_{2j}^{i+1}. Subsequently, we subtract the corresponding multisets Φ_{i+1} and $\neg\Phi_{i+1}$, respectively. In other words, $\alpha_j^i \cdot \Theta_j^i$ is replaced by two multisets: $\alpha_{2j-1}^{i+1} \cdot \Theta_{2j-1}^{i+1}$ and $\alpha_{2j}^{i+1} \cdot \Theta_{2j}^i$.

Intuitively, when $1 \le j \le 2^i$, the multiset Θ_j^i obtained by rule application j represents an interpretation for the variables $x_1, \ldots, x_i$ ranging from when all the variables are true to when all of them are false. Let these interpretations be denoted as $\mathcal{I}_1, \ldots, \mathcal{I}_{2^i}$ for a fixed $1 \le i \le n$. Then Θ_j^i comprises the t_p's such that $\mathcal{I}_j$ does not satisfy D_p, where $1 \le j \le 2^i$ and $1 \le p \le m$. We perform all these rewrite steps for variables $x_1, \ldots, x_n$, resulting in a total of n rewrite steps.

Regarding our running example, we can establish the following rules for the calculus

$$\begin{aligned}
\alpha \mid x_1 \mid \Delta \to & \alpha_1^1 \cdot \Upsilon \ominus_\Lambda \alpha_1^1 \cdot \Psi_1 \mid \\
& \alpha_2^1 \cdot \Upsilon \ominus_\Lambda \alpha_2^1 \cdot \neg\Psi_1 \mid \Delta, \\
\{|\alpha_1^1|\}_X \mid \{|\alpha_2^1|\}_Y \mid x_2 \mid \Delta \to & \{|\alpha_1^2|\}_X \ominus_\Lambda \alpha_1^2 \cdot \Psi_2 \mid \\
& \{|\alpha_2^2|\}_X \ominus_\Lambda \alpha_2^2 \cdot \neg\Psi_2 \mid \\
& \{|\alpha_3^2|\}_Y \ominus_\Lambda \alpha_3^2 \cdot \Psi_2 \mid \\
& \{|\alpha_4^2|\}_Y \ominus_\Lambda \alpha_4^2 \cdot \neg\Psi_2 \mid \Delta,
\end{aligned}$$

where $\Upsilon = t_1 \mid t_2$ and $\Delta = \beta_1 \cdot \Psi_1 \mid \beta_2 \cdot \Psi_2 \mid \neg\beta_1 \cdot \neg\Psi_1 \mid \neg\beta_2 \cdot \neg\Psi_2$. The set of rules is determined solely by the number of variables and the number of conjuncts in F and can be constructed independently of the other properties of F.

We now need to verify whether, after performing n rewrite steps, at least one of the multisets $\alpha_j^n \cdot \Theta_j^n$ $(1 \le j \le 2^n)$ is actually empty, denoted as Λ in our case. To this end, we define the following reduction rule:

$$\Lambda \mid \widetilde{x}_2 \mid \ldots \mid \widetilde{x}_{2^n} \mid \widetilde{y}_1 \mid \ldots \mid \widetilde{y}_n \mid \widetilde{y}_{n+1} \mid \ldots \mid \widetilde{y}_{2n} \to t$$

The rule above unifies with the result if and only if at least one of the strings is Λ. The variables $\widetilde{x}_2, \ldots, \widetilde{x}_{2^n}$ and $\widetilde{y}_1, \ldots, \widetilde{y}_n, \widetilde{y}_{n+1}, \ldots, \widetilde{y}_{2n}$ stand for the rest of the strings. These are of the form $\alpha_j^n \cdot \Theta_j^n$ $(1 \le j \le 2^n)$, or $\beta_k \cdot \Phi_k$, or $\neg\beta_k \cdot \neg\Phi_k$ $(1 \le k \le n)$, or Λ, respectively. Hence, if Λ occurs among the multisets representing the interpretations for the variables $x_1, \ldots, x_n$, then the entire computation evaluates to t. This determination can be made with a single rule application due to the commutativity of the multiset constructor.

In the context of our example, this implies that we need to supplement the set of rules with the following rule

$$\Lambda \mid \widetilde{x}_2 \mid \widetilde{x}_3 \mid \widetilde{x}_4 \mid \widetilde{y}_1 \mid \widetilde{y}_2 \mid \widetilde{y}_3 \mid \widetilde{y}_4 \to t.$$

This results in the following reduction sequence

$$\begin{aligned} \alpha \mid x_1 \mid x_2 \mid \Xi &\rightarrow \\ \alpha_1^1 \cdot t_3 \mid \alpha_2^1 \cdot (t_1 \mid t_2) \mid x_2 \mid \Xi &\rightarrow \\ \alpha_1^2 \cdot t_3 \mid \Lambda \mid \alpha_3^2 \cdot t_2 \mid \alpha_4^2 \cdot t_1 \mid \Xi &\rightarrow \\ t, & \end{aligned}$$

where $\Xi = \beta_1 \cdot (t_1 \mid t_2) \mid \beta_2 \cdot t_1 \mid \neg\beta_1 \cdot t_3 \mid \neg\beta_2 \cdot (t_2 \mid t_3)$, showing that F is satisfiable. □

Remark 11 Assume a CNF $C = D_1 \wedge D_2 \wedge \ldots D_m$ with n variables and m conjuncts is given. The above proof presents a string multiset rewriting calculus with $n + 1$ rewriting rules and also a multiset of strings M_C. By applying the rules successively to M_C, we are able to decide in $n + 1$ rewriting steps whether C is satisfiable. In total, $n + 1$ rule applications are needed to achieve that goal. The multiset M_C can be constructed in time polynomial with respect to n and m.

We have to make several observations at this point. First of all, we observe that our solution is universal in the sense that, although M_C is specific to C, the rewriting steps are the same for every CNF with n variables and m conjuncts and there are $n + 1$ of them. Each rule is applied exactly once for the emerging multiset in the decision process. The result, however, will be connected with C since the rewriting steps involve substitutions for the variables in the multiset patterns. Notably, the multisets Φ_i and $\neg\Phi_i$ encode the appearance of variable x_i in the conjuncts for C ($1 \leq i \leq n$), hence, different initial CNFs can lead to different outcomes.

Secondly, if we determine the length of the rules by the sum of the cardinalities of their left hand side and right hand side, there are rules of length proportional to 2^n, where n is the number of variables in C. For this reason, the assumption that an application of a rule takes place in constant time is crucial for ensuring the efficiency of the model.

Once constructed, the rules will be universally applicable for all CNFs with n variables and m conjuncts. This is illustrated in the following example.

In the theorem above, we examined the satisfiability of the conjunctive normal form F, with $n = 2$ variables and $m = 3$ conjuncts. We asserted that, when n and m have the same values, the constructed rules are universal for every CNF of that form.

To conclude this section, we provide an example to illustrate this universality of the model.

Example 12 In this example, we examine the satisfiability of the CNF $G = (x_1 \vee x_2) \wedge (\neg x_1 \vee x_2) \wedge \neg x_2$. In this case, $n = 2$ and $m = 3$ allowing us to establish the following rules already determined in the proof of the theorem

$$\begin{aligned}
\alpha \mid x_1 \mid \Delta \rightarrow & \alpha_1^1 \cdot \Upsilon \ominus_\Lambda \alpha_1^1 \cdot \Psi_1 \mid \\
& \alpha_2^1 \cdot \Upsilon \ominus_\Lambda \alpha_2^1 \cdot \neg\Psi_1 \mid \Delta, \\
\{|\alpha_1^1|\}_X \mid \{|\alpha_2^1|\}_Y \mid x_2 \mid \Delta \rightarrow & \{|\alpha_1^2|\}_X \ominus_\Lambda \alpha_1^2 \cdot \Psi_2 \mid \\
& \{|\alpha_2^2|\}_X \ominus_\Lambda \alpha_2^2 \cdot \neg\Psi_2 \mid \\
& \{|\alpha_3^2|\}_Y \ominus_\Lambda \alpha_3^2 \cdot \Psi_2 \mid \\
& \{|\alpha_4^2|\}_Y \ominus_\Lambda \alpha_4^2 \cdot \neg\Psi_2 \mid \Delta, \\
\Lambda \mid \widetilde{x}_2 \mid \widetilde{x}_3 \mid \widetilde{x}_4 \mid \widetilde{y}_1 \mid \widetilde{y}_2 \mid \widetilde{y}_3 \mid \widetilde{y}_4 \rightarrow & t,
\end{aligned}$$

where $\Upsilon = t_1 \mid t_2$ and $\Delta = \beta_1 \cdot \Psi_1 \mid \beta_2 \cdot \Psi_2 \mid \neg\beta_1 \cdot \neg\Psi_1 \mid \neg\beta_2 \cdot \neg\Psi_2$. Regarding the multiset, we have $\Phi_1 = t_1, \neg\Phi_1 = t_2, \Phi_2 = t_1 \mid t_2, \neg\Phi_2 = t_3$. By which, $M_G = \alpha \mid x_1 \mid x_2 \mid \beta_1 \cdot t_1 \mid \beta_2 \cdot (t_1 \mid t_2) \mid \neg\beta_1 \cdot t_2 \mid \neg\beta_2 \cdot t_3$ follows. We obtain the reduction sequence as below.

$$\begin{aligned}
\alpha \mid x_1 \mid x_2 \mid \Xi \rightarrow \\
\alpha_1^1 \cdot (t_2 \mid t_3) \mid \alpha_2^1 \cdot (t_1 \mid t_3) \mid x_2 \mid \Xi \rightarrow \\
\alpha_1^2 \cdot t_3 \mid \alpha_2^2 \cdot t_2 \mid \alpha_3^2 \cdot t_3 \mid \alpha_4^2 \cdot t_1 \mid \Xi,
\end{aligned}$$

where $\Xi = \beta_1 \cdot t_1 \mid \beta_2 \cdot (t_1 \mid t_2) \mid \neg\beta_1 \cdot t_2 \mid \neg\beta_2 \cdot t_3$. The last multiset cannot be reduced any further, as none of the strings equals Λ. Hence, we can deduce that G is not satisfiable.

4 Conclusion

In this paper, we selected the String Multiset Rewriting calculus, developed by Barbuti et al [4], as our computational model. This calculus operates on multisets of strings, utilizing a rewrite mechanism based on pattern matching. We aimed to demonstrate that the model could, at least in principle, serve as a computational tool for solving hard problems. We achieved this by showing that, for any conjunctive normal form $C = D_1 \vee \ldots \vee D_m$ of n variables, there exists a multiset comprised of $3n + 1$ strings, each string being of length at most m, and there are $n + 1$ rewrite rules such that the subsequent application of the rewrite rules determines the satisfiability of C in $n + 1$ steps. Importantly, this method is universal: the rewrite rules apply uniformly to any conjunctive normal form of n variables and m conjuncts. Only the starting multiset may vary, indicating the difference. This demonstrates that the String Multiset Rewriting calculus could potentially serve as a parallel computational model, significantly reducing computational time once an efficient implementation is provided.

Funding Open access funding provided by University of Debrecen. The work is supported by the EFOP-3.6.3-VEKOP-16-2017-00002 project.

Declarations

Conflict of interest The author declares no conflict of interests.

References

1. Alhazov A, Cojocaru S, Gheorghe M, Rogozhin Y, Rozenberg G, Salomaa A (eds.) (2014) Membrane Computing. CMC 2013. Lecture Notes in Computer Science, vol. 8340, Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-54239-8_14
2. Ausiello G, Karhumäki J, Mauri G, Ong L (eds.) (2008) Fifth IFIP international conference on theoretical computer science-TCS 2008. IFIP International federation for information processing, Springer, vol. 273, Boston, MA. https://doi.org/10.1007/978-0-387-09680-3_18
3. Bagossy A, Battyányi P, An encoding of the λ-calculus in the String MultiSet Rewriting calculus, Acta Informatica (to appear)
4. Barbuti R, Caravagna G, Maggiolo-Schettini A, Milazzo P (2008) An intermediate language for the simulation of biological systems. Electron Notes Theoret Comput Sci 194:19–34
5. Barbuti R, Maggiolo-Schettini A, Milazzo P, Troina A (2006) A calculus of looping sequences for modelling microbiological systems. Fund Inform 72:21–35
6. Braich RS, Chelyapov N, Johnson C, Rothemund PWK, Adleman L (2002) Solution of a 20-Variable 3-SAT problem on a DNA computer. Science 296:499–502. https://doi.org/10.1126/science.1069528
7. Cardelli L, Brane Calculi. Interactions of biological membranes. In [10] 257–280
8. Cardelli L (2008) From processes to ODEs by Chemistry. In [2] 261–281
9. Danos V, Laneve C (2004) Formal molecular biology. Theoret Comput Sci 325:69–110
10. Danos V, Schachter V (eds.) (2005) 1CMSB'04: Proceedings of the 20 international conference on computational methods in systems biology. Lecture Notes in Computer Science, vol. 3082, Springer, Berlin
11. Gazdag Zs Solving SAT by P Systems with active membranes in linear time in the number of variables. In [1] 189–205
12. Gutiérrez-Naranjo MA, Pérez-Jiménez MJ, Romero-Campero FJ (2007) A uniform solution to SAT using membrane creation. Theoret Comput Sci 371:54–61
13. Lipton RJ (1995) Using DNA to solve NP-complete problems. Science 268:542–545
14. Martinelli F, Bistarelli S, Cervesato I, Lenzini G, Marangoni R, Representing biological systems through multiset rewriting. In [15], 415–426
15. Moreno Diaz R, Pichler F (eds.) (2004) Computer aided systems theory (EUROCAST '03). Lecture Notes in Computer Science, vol. 2809, Springer
16. Păun Gh (1999) P systems with active membranes: attacking NP complete problems. J Automata Lang Comb 6(1):75–90
17. Păun Gh (2000) Computing with membranes. J Comput Syst Sci 61(1):108–143
18. Priami C (1995) Stochastic π-Calculus. Comput J 38(7):578–589
19. Thachuk C, Liu Y (eds.) (2019) DNA computing and molecular programming. DNA 2019. Lecture Notes in Computer Science, vol. 11648, Springer, Cham. https://doi.org/10.1007/978-3-030-26807-7_1
20. Troják M, Šafránek D, Brim L, Šnalagovič J, Červený J (2020) Executable biochemical space for specification and analysis of biochemical systems. Electron Notes Theoret Comput Sci 350:91–116

21. Troják M, Pastva S, Šafránek D, Brim L, Regulated multiset rewriting systems, arXiv:2111.13036
22. Troják M, Šafránek D, Pastva S, Brim L (2023) Rule-based modelling of biological systems using regulated rewriting. Biosystems 225:104843. https://doi.org/10.1016/j.biosystems.2023.104843
23. Winfree E. Chemical reaction networks and stochastic local search. In [19] 1–20