Generating Pairwise Disjoint Families through DNA Computations

MARIO J. PÉREZ-JIMÉNEZ
FERNANDO SANCHO-CAPARRINI

Dpto. Ciencias de la Computación e Inteligencia Artificial
Universidad de Sevilla, E.T.S. Ingeniería Informática
Avda. Reina Mercedes, s/n – 41012 Sevilla, Spain
E-mail: {Mario.Perez,Fernando.Sancho}@cs.us.es

Abstract

In this paper we study how to generate pairwise disjoint families from a finite collection of given sets in a molecular model with random access memory: the *sticker model*. We analyze problems related to the formal verification of a molecular program in this kind of models. The invariant techniques we use are based on a labelling of tubes and followed by a detailed study of the evolution of each molecule from the initial tube along the execution. The results are applied to obtain molecular solutions of two well-known numerical **NP**-complete problems: the *Exact Cover* problem and the *Set Packing* problem.

M. J. Pérez-Jiménez et al. (Eds.): Recent Results in Natural Computing, pp. 231–246
© The Authors © Fénix Editora

1 Introduction

The computational solvability of **NP**-complete problems in practice has achieved a quantitative improvement with the born of DNA computing at the end of 1994. The arise of the first molecular models, in the beginning of 1995, and the universality of these models (i.e., with the computational power of Turing Machines) allow, at least in a theoretical way, to deal with the solvability of any **NP**-complete problem through DNA computations in an efficient manner.

Solving a problem, X, in a molecular model consists in the design of a program, P, within this model, that can solve it in the following sense: for each instance, E, of the problem X, the execution of the program P over an initial tube encoding all possible solutions associated with E, produces a tube (or a set of tubes) encoding the solutions to the problem P respect to the input data E. Verifying (X, P) consists in proving that the program P solves, in fact, the problem X; that is, to verify (X, P) we have to follow two steps: (a) to prove that every molecule of the output tube encodes a correct solution of the problem (*Soundness* of the program); (b) to prove that every molecule of the input tube encoding a correct solution of the problem is in the output tube (*Completeness* of the program).

The goal of this paper is the design, analysis and formal verification of a molecular program within the sticker model solving the problem of generating pairwise disjoint subfamilies of a finite collection of finite sets. The paper begins by briefly introducing a sticker based model for DNA computations. In section 3 we outline a problem, then a molecular program is designed in this model to solve the problem and finally we analyze the time complexity (number of molecular operations) and the space complexity (size of the input tube and total number of tubes used along the execution) of the program. The main contribution of this work is presented in section 4, where a formal verification of the designed program is given. In section 5 we apply the above results to solve two well-known **NP**-complete numerical problems: the *Exact Cover* problem and the *Set Packing* problem [1].

2 A Sticker-Based Model

The *sticker model* was introduced by S. Roweis et al [3] as a model of molecular computation making use of DNA strands as the physical substrate in which information is represented. This model has a random access memory, where no strand extension is required, and uses no enzymes. The materials are reusable, at least in theory.

We first describe a way, based on complementarity, of representing information by DNA molecules. The *sticker model* uses two basic groups of single stranded DNA molecules in its representation of a bit string, referred to as *memory strands* and *sticker strands*.

An (n,k,m)-*memory strand*, with $n \geq k \cdot m$, consists of a strand of n bases subdivided into k non-overlapping substrands each of which is m bases long. The substrands should be significantly different from one another: any two of them should differ with respect to several base positions.

A *sticker* associated with an (n,k,m)-memory strand is m bases long and complementary to exactly one of the k substrands in the memory strand. A specific substrand of a memory strand can be either *on* or *off*. If a sticker is annealed to its matching substrand on a memory strand, then the particular substrand is said to be *on*, if not it is said to be *off*.

An (n,k,m)-*memory complex* is an (n,k,m)-memory strand along with its annealed stickers (if any). Memory complexes are DNA strands that are partially complemented, and represent binary numbers, where a substring being *on* (respectively, *off*) represents the bit 1 (respectively, 0). So, (n,k,m)-memory complexes can represent bit strings of $\{0,1\}^k$, for this reason, it is usual to identify them either as binary functions ($\sigma : \{1,...,k\} \longrightarrow \{0,1\}$, such that $\sigma(i)=1$ if and only if the ith substrand is *on*), or as subsets of $\{1,...,k\}$ by means of their characteristic functions.

We are now ready to introduce the *basic operations* used in the sticker model. A *test tube* or shortly a *tube*, is a finite multiset (a collection where elements can be repeated) of memory complexes. The operations over tubes in the sticker model we consider in this paper are the following:

- `Combine`(T_1,T_2): given two test tubes T_1 and T_2 this operation produces a new tube, $T_1 \cup T_2$, being the multiset union of T_1 and T_2. Thus, by this operation two test tubes are combined into one.

- `Separate`(T, i): given a test tube T and a natural number i, with $1 \leq i \leq k$, this operation produces two new tubes, $+(T,i) = \{\{\sigma \in T : \sigma(i) = 1\}\}$ and $-(T,i) = \{\{\sigma \in T : \sigma(i) = 0\}\}$.

  That is, the test tube $+(T,i)$ (respectively, $-(T,i)$) consists of all the memory complexes in the original tube T where the ith substrand is *on* (respectively, *off*).

  We usually write $(T_1, T_2) \leftarrow$ `separate`(T, i) to indicate that $T_1 = +(T,i)$ and $T_2 = -(T,i)$.

- `Set`(T, i): given a test tube T and a natural number i, with $1 \leq i \leq k$, this operation produces a new tube, `Set`(T, i), where the ith substrand of each memory complex in T is turned *on*. That is, an appropriate sticker is annealed to it if the ith substrand is *off* in the memory complex, but the ith substrand is left unchanged if it is already annealed.

- `Read`(T): given a test tube T this operation reads the content of tube T. To achieve that, one memory complex has to be isolated from T and its annealed stickers determined, or else it has to be reported that the tube T contains no memory complexes.

Note that, in this model, the operations `Separate` and `Set` are the only ones implementing a massive parallelism. Also, only operation `Set` can modify the inner structure of the molecules of a tube.

The input or *initial test tube* will be a *library* of memory complexes. In particular, a (k,l)-*library*, with $1 \leq l \leq k$, consists of memory complexes with k substrands, where the first l substrands are either *on* or *off*, in all possible ways, whereas the last $k - l$ substrands are *off*.

Computations in the sticker model consist of a finite sequence of the operations mentioned above, that can be written in a simple way by means of robotic operations (such as loops, conditionals, etc.).

If a molecular program P has a main loop FOR or $WHILE$, then we can establish the soundness and completeness of it by means of searching invariant formulas (over the variables in the program) of the loop with inductive techniques. First, we proceed to a labelling of the tubes to individualize them along the execution. Second, having in mind that the inner structure can be modified in the sticker model, an itemized study of the evolution of every molecule along the execution of the program will be indispensable. This will be implemented through a function, denoted by $STEP$.

3 Pairwise Disjoint Families

The *Pairwise Disjoint Families* problem is the following: *Let* $A = \{1, \dots, p\}$. *Let* $\mathcal{F} = \{B_1, \dots, B_q\}$ *a finite family of subsets of* A. *Determine all pairwise disjoint subfamilies of* $\mathcal{F}$.

We will denote $r_i = |B_i|$ and $B_i = \{x_i^1, \dots, x_i^{r_i}\}$, for each i such that $1 \leq i \leq q$. The design of the program can be summarized as follows: start from an initial test tube, T_0, whose molecules encode all possible subfamilies of $\mathcal{F}$ (for example, T_0 is a $(p+q, q)$-library); we separate all molecules of T_0 with respect to the position 1 to produce the tubes T^+ and T^-; from tube T^+ we remove the molecules with the $(q + x_1^j)$th position set to 1 (for some $x_1^j \in B_1$), and we set to 1 all $(q + x_1^j)$th positions, for each j such that $1 \leq j \leq r_1$, in the remaining molecules; we make $T_0 = T_{r_1}^* \cup T^-$ and repeat the process for positions from 2 to q (with respect to the sets B_2 to B_q, respectively).

Keeping in mind the above idea we design the following molecular program within the sticker model, for finding a pairwise disjoint subfamilie in any finite family:

```
Procedure Disjoint
Input:  $T_0$ (a $(p+q,q)$-library)
      for $i=1$ to $q$ do
           $(T^+, T^-) \leftarrow$ separate$(T_0, i)$
           $T_0^* \leftarrow T^+$
           for $j=1$ to $r_i$ do
                $(T^{trash}, T) \leftarrow$ separate$(T_{j-1}^*, q + x_i^j)$
                $T_j^* \leftarrow$ set$(T, q + x_i^j)$
           end for
           $T_0 \leftarrow$ combine$(T_{r_i}^*, T^-)$
      end for
```

This program executes $q + r$ `separate` operations, r `set` operations and q `combine` operations, where $r = r_1 + \cdots + r_q$. Hence the number of molecular operations of this program is $2q+2r \in O(q \cdot p)$. The number of used tubes is of the order of $O(p)$. The number of molecules of the input tube is of the order of 2^q.

4 Formal Verification

In order to establish a formal verification of this program we begin with a procedure for the labelling of the tubes:

```
Procedure Disjoint
Input: $T_0$
     for $i = 1$ to $q$ do
          $(T_i^+, T_i^-) \leftarrow$ separate$(T_{i-1}, i)$
          $T_{i,0}^* \leftarrow T_i^+$
          for $j = 1$ to $r_i$ do
               $(T_{i,j}^{trash}, T_{i,j}^{\bullet}) \leftarrow$ separate$(T_{i,j-1}^*, q + x_i^j)$
               $T_{i,j}^* \leftarrow$ set$(T_{i,j}^{\bullet}, q + x_i^j)$
          end for
          $T_i \leftarrow$ combine$(T_{i,r_i}^*, T_i^-)$
     end for
```

Notation: Let $\sigma \in T_0$. Then the molecule σ is represented by a binary string of 0's and 1's with length $q + p$. Hence we can consider σ as the range of the characteristic function of a subset $B(\sigma)$ (we write B for short) of $\{1, \ldots, q + p\}$. That is, we can identify the molecule σ with the subset

$$B = \{i : 1 \leq i \leq q + p \wedge \sigma(i) = 1\}$$

For each i such that $1 \leq i \leq q$, let $B_i = \{x_i^1, \ldots, x_i^{r_i}\}$. We denote by $q + B_i$ the set $\{q + x_i^j : 1 \leq j \leq r_i\}$. That is, $\sigma \cap (q + B_i) = \emptyset$ means that $\forall j \ (1 \leq j \leq r_i \Rightarrow \sigma(q + x_i^j) = 0)$.

If $\sigma \in T_0$ then we denote by σ_q the substring $\sigma(1) \ldots \sigma(q)$ and we denote by σ_p the substring $\sigma(q + 1) \ldots \sigma(q + p)$. That is, the string σ_q encodes a subset of $\{1, \ldots, q\}$ that we can identify with a subfamily of $\mathcal{F} = \{B_1, \ldots, B_q\}$, and the string σ_p encodes a subset of $A = \{1, \ldots, p\}$. That is,

$$\sigma_q \equiv \{B_i \in \mathcal{F} : 1 \leq i \leq q \wedge \sigma(i) = 1\}$$

$$\sigma_p \equiv \{j : 1 \leq j \leq p \wedge \sigma(j) = 1\}$$

Next we define a function, namely $STEP$, that captures the evolution of each molecule after the execution of one step of the main loop. Note that there are molecules removed after the execution of some steps in the main loop (they go into tube $T_{i,j}^{trash}$). So, in general, the function $STEP$ is not necessarily a total function.

Definition 1. Let i be such that $1 \leq i \leq q$. Let $\rho \in T_{i-1}$. We define

$$STEP(\rho, i) = \begin{cases} \rho & \text{if } i \notin \rho \\ \rho \cup (q + B_i) & \text{if } i \in \rho \text{ and } \rho \cap (q + B_i) = \emptyset \\ \uparrow & \text{otherwise} \end{cases}$$

In the first two cases we will write $STEP(\rho, i) \downarrow$. Besides, we define $STEP_i(\rho) = STEP(\rho, i)$.

Note that if $STEP(\rho, i) \downarrow$, then $\rho \subseteq STEP(\rho, i)$ and that $STEP_i$ is the identity function on T_i^-. Now, let us see that $STEP_i$ is a partial function from T_{i-1} to T_i.

Lemma 1. *Let i be such that $1 \leq i \leq q$. Then for every molecule $\rho \in T_{i-1}$ such that $STEP(\rho, i) \downarrow$ we have $STEP(\rho, i) \in T_i$.*

Proof. Let i be such that $1 \leq i \leq q$. Let $\rho \in T_{i-1}$ such that $STEP(\rho, i) \downarrow$.

If $i \notin \rho$, then $\rho \in -(T_{i-1}, i) = T_i^- \subseteq T_i$, and the result follows from the definition of $STEP$. If $i \in \rho$ and $\rho \cap (q + B_i) = \emptyset$, then we can define recursively $\rho_{(j)}$, for each j such that $0 \leq j \leq r_i$, as follows:

$$\begin{cases} \rho_{(0)} = \rho \\ \rho_{(j+1)} = \rho_{(j)} \cup \{q + x_i^{j+1}\} \end{cases}$$

Let us see that $\forall\, j\ (0 \leq j \leq r_i \Rightarrow \rho_{(j)} \in T_{i,j}^*)$, by induction on j.

- We have $\rho_{(0)} = \rho \in +(T_{i-1}, i) = T_i^+ = T_{i,0}^*$. Let $j < r_i$ such that $\rho_{(j)} \in T_{i,j}^*$. From definition of $\rho_{(j)}$ it follows that $\forall\, t\ (j < t \leq r_i \Rightarrow \rho_{(j)}(t) = \rho(t))$. Hence, $\rho_{(j)}(q + x_i^{j+1}) = \rho(q + x_i^{j+1}) = 0$. That is, $\rho_{(j)} \in -(T_{i,j}^*, q + x_i^{j+1}) = T_{i,j+1}^*$. But $\rho_{(j+1)} = \rho_{(j)}$ except by $\rho_{(j)}(q + x_i^{j+1}) = 0$ and $\rho_{(j+1)}(q + x_i^{j+1}) = 1$. Then $\rho_{(j+1)} \in \texttt{set}\ (T_{i,j+1}^{\bullet}, q + x_i^{j+1}) = T_{i,j+1}^*$.

Taking into account that $\rho_{(r_i)} = STEP(\rho, i)$ from the above definition, we conclude that $STEP(\rho, i) \in T_{i,r_i}^* \subseteq T_i$. $\square$

The execution of the designed molecular program can be interpreted as the evolution over time of a population consisting, at the beginning, of a multiset of molecules in the input tube (every molecule represents an element, thus, in the original population cloned members can exist). Every step of the main loop takes one unit of time. Hence, throughout the execution an element of the population can die (that is, it can be thrown away) or it can survive. If the molecular computing model has random access memory (that is, if the inner structure of the molecules can be modified) then the molecules surviving all the steps can be of two kinds: those whose inner structure have been modified and those that remain with no modifications.

The above Lemma allows us to define the *history* of every molecule in the input tube T_0.

Definition 2. For every molecule $\sigma \in T_0$ we define

$$\begin{cases} \sigma^0 = \sigma \\ \sigma^{i+1} = STEP(\sigma^i, i+1), \text{ if } 0 \leq i < q \end{cases}$$

Let $\sigma \in T_0$ such that there exists i, with $1 \leq i \leq q$, verifying $\sigma^i \downarrow$. Then we will say that the molecule σ has *survived* after the execution of ith step of the main loop.

Next, we provide a technical lemma that will allow us to make a formal verification, with a short outline of its proof. Let us recall that for each i such that $1 \leq i \leq q$, $B_i = \{x_i^1, \ldots, x_i^{r_i}\}$. Then, for each i, j such that $1 \leq i \leq q$ and $1 \leq j \leq r_i$, we denote by B_i^j the set $\{x_i^1, \ldots, x_i^j\}$.

Lemma 2. $\forall\, i\ (0 \leq i \leq q \rightarrow \forall\, \sigma \in T_0\ (\sigma^i \downarrow \rightarrow \sigma^i \in T_i))$.

Proof. By induction on i. The case $i = 0$ is obvious. Let i be such that $i < q$ and assume that $\forall\, \sigma \in T_0\ (\sigma^i \downarrow \rightarrow \sigma^i \in T_i)$ holds. Then we will prove the result for $i+1$. For that, let $\sigma \in T_0$ such that $\sigma^{i+1} \downarrow$. Then $STEP(\sigma^i, i+1) \downarrow$. So, $\sigma^i \downarrow$. From the induction hypothesis we have $\sigma^i \in T_i$. Then, we conclude from Lemma 1 that $STEP(\sigma^i, i+1) \in T_{i+1}$. Hence, $\sigma^{i+1} \in T_{i+1}$. □

Lemma 3. *For each i, j such that $1 \leq i \leq q$ and $1 \leq j \leq r_i$ we have the following:*

$$\forall\, \tau \in T_{i,j}^{*}\ \exists\, \rho \in T_i^{+}\ (\tau = \rho \cup (q + B_i^j)\ \wedge\ \rho \cap (q + B_i^j) = \emptyset))$$

Proof. Let i be such that $1 \leq i \leq q$. Let us see that

$$\forall j\, (1 \leq j \leq r_i \rightarrow \forall \tau \in T^*_{i,j}\ \exists \rho \in T^+_i\ (\tau = \rho \cup (q+B^j_i) \wedge \rho \cap (q+B^j_i) = \emptyset))$$

By induction on j. Let $\tau \in T^*_{i,1} = \texttt{set}(T^\bullet_{i,1}, q + x^1_i)$. Then there exists $\rho \in T^\bullet_{i,1}$ such that $\rho = \tau \cup (q + B^1_i)$. Since $\rho \in T^\bullet_{i,1} = -(T^*_{i,0}, q + x^1_i)$, it follows that $\rho \in T^*_{i,0} = T^+_i$, and $\rho(q + x^1_i) = 0$. That is, the result holds for $j = 1$.

Let us assume the result holds for j such that $j \geq 1$ and $j < r_i$. Let $\tau \in T^*_{i,j+1} = \texttt{set}(T^\bullet_{i,j+1}, q + x^{j+1}_i)$. Then there exists $\rho' \in T^\bullet_{i,j+1}$ such that $\rho' = \tau \cup (q + B^{j+1}_i)$. Since $\rho' \in T^\bullet_{i,j+1} = -(T^*_{i,j}, q + x^{j+1}_i)$, it follows that $\rho' \in T^*_{i,j} = T^+_i$, and $\rho'(q + x^{j+1}_i) = 0$.

Taking into account that $\rho' \in T^*_{i,j}$, from the induction hypothesis we deduce that there exists $\rho \in T^+_i$ such that $\rho' = \rho \cup (q+B^j_i) \wedge \rho \cap (q+B^j_i) = \emptyset$. So, we conclude that $\tau = \rho \cup (q + B^j_i)\ \wedge\ \rho \cap (q + B^{j+1}_i) = \emptyset$. □

Lemma 4. *For each i such that $1 \leq i \leq q$ we have the following: for every molecule $\tau \in T^*_{i,r_i}$ there exists a molecule $\rho \in T_{i-1}$ verifying $i \in \rho\ \wedge\ \rho \neq \tau\ \wedge\ STEP(\rho, i) = \tau$.*

Proof. Let i be such that $1 \leq i \leq q$. Let $\tau \in T^*_{i,r_i}$. From Lemma 3, with $j = r_i$, we deduce that there exists $\rho \in T^+_i$ such that $\tau = \rho \cup (q + B^j_i)\ \wedge\ \rho \cap (q + B^j_i) = \emptyset$. Then $\rho \neq \tau$ and $\rho \in T^+_i = +(T_{i-1}, i)$. That is, $\rho \in T_{i-1}$ and $i \in \rho$. Hence, $STEP(\rho, i) = \tau$. □

Lemma 5. $\forall i\, \forall j\, (1 \leq i \leq q\ \wedge\ 1 \leq j \leq r_i \implies T^*_{i,j} \cap T^-_i = \emptyset)$.

Proof. Let i, j be such that $1 \leq i \leq q$ and $1 \leq j \leq r_i$. Let $\tau \in T^*_{i,j}$. From Lemma 3, we deduce that there exists $\rho \in T^+_i = +(T_{i-1}, i)$ such that $\tau = \rho \cup (q + B^j_i)$ and $\rho \cap (q + B^j_i) = \emptyset$. Then $\rho \in T_{i-1}$, $i \in \rho$, and $\tau_q = \rho_q$. Hence, $\tau \notin -(T_{i-1}, i)$. That is, $\tau \notin T^-_i$. □

Lemma 6. *For each i such that $1 \leq i \leq q$, and for every molecule $\tau \in T^-_i$ we have $\tau \in T_{i-1}$ and $STEP(\tau, i) = \tau$.*

Proof. It follows easily from the definition of function STEP, and taking into account that $T^-_i = -(T_{i-1}, i)$. □

Lemma 7. *For each i such that $1 \leq i \leq q$, and for every molecule $\rho, \tau \in T_{i-1}$ such that $STEP(\rho, i) \downarrow= STEP(\tau, i)$, we have $\rho = \tau$.*

Proof. Let i be such that $1 \leq i \leq q$ and let $\rho, \tau \in T_{i-1}$ be such that $STEP(\rho, i) \downarrow= STEP(\tau, i)$. Then $\rho_q = (STEP(\rho, i))_q = (STEP(\tau, i))_q = \tau_q$.

- If $i \notin \rho$ then $i \notin \tau$. Thus $\rho = STEP(\rho, i) = STEP(\tau, i) = \tau$.
- If $i \in \rho$ and $\rho \cap (q + B_i) = \emptyset$ then $i \in \tau$ and $\tau \cap (q + B_i) = \emptyset$. Let $\rho' = STEP(\rho, i)$ and $\tau' = STEP(\tau, i)$. Then $\rho' = \rho \cup (q + B_i)$ and $\tau' = \tau \cup (q + B_i)$. From $\rho' = \tau'$ it follows that $\rho = \tau$.

□

Lemma 8. $\forall i\, (0 \leq i \leq q \rightarrow \forall \sigma\, \forall \rho \in T_0\, (\sigma^i \downarrow= \rho^i \rightarrow \sigma = \rho))$.

Proof. Let us prove the result by induction on i. The result is obviously true for $i = 0$. Assume that it is true for $i < q$. Let $\sigma, \rho \in T_0$ such that $\sigma^{i+1} \downarrow= \rho^{i+1}$. Then $STEP(\sigma^i, i+1) \downarrow= STEP(\rho^i, i+1)$. From Lemma 7 we obtain $\sigma^i \downarrow= \rho^i$. Finally, from the induction hypothesis it follows that $\sigma = \rho$. □

Lemma 9. $\forall i\, (0 \leq i \leq q \rightarrow \forall \tau \in T_i\, \exists ! \sigma \in T_0\, (\sigma^i = \tau))$.

Proof. Let us first prove the existence by induction on i. The case $i = 0$ is obvious.

Let us assume the result holds for $i < q$. Let $\tau \in T_{i+1} = T^*_{i+1, r_{i+1}} \cup T^-_{i+1}$. If $\tau \in T^*_{i+1, r_{i+1}}$, from Lemma 4 it follows that there exists a molecule $\rho \in T_i$ such that $i + 1 \in \rho \wedge \rho \neq \tau \wedge STEP(\rho, i+1) = \tau$. As $\rho \in T_i$, from the induction hypothesis we deduce that there exists $\sigma \in T_0$ such that $\rho = \sigma^i$. Hence, $\tau = STEP(\rho, i+1) = STEP(\sigma^i, i+1) = \sigma^{i+1}$.

If $\tau \in T^-_{i+1}$, from Lemma 6 we have $\tau \in T_i$ and $STEP(\tau, i+1) = \tau$. As $\tau \in T_i$, from the induction hypothesis we deduce that there exists a molecule $\sigma \in T_0$ such that $\tau = \sigma^i$. Hence, $\tau = STEP(\tau, i+1) = STEP(\sigma^i, i+1) = \sigma^{i+1}$.

The uniqueness of such a molecule is obtained from Lemma 8. □

Lemma 10. $\forall i\, (0 \leq i \leq q \rightarrow \forall \sigma \in T_0\, (\sigma^{i+1} \downarrow\rightarrow \sigma^i_q = \sigma^{i+1}_q \wedge \sigma^i_p \subseteq \sigma^{i+1}_p))$.

Proof. Let i be such that $1 \leq i \leq q$. Let $\sigma \in T_0$ such that $\sigma^{i+1} \downarrow$. From Lemma 2 we have $\sigma^{i+1} \subseteq T_{i+1}$. Taking into account that $T_{i+1} = T^*_{i+1, r_{i+1}} \cup T^-_{i+1}$, we distinguish two cases.

- If $\sigma^{i+1} \in T^*_{i+1,r_{i+1}}$, from Lemma 4 we deduce that there exists $\rho \in T_i$ verifying $i+1 \in \rho \wedge \rho \neq \sigma^{i+1} \wedge STEP(\rho, i+1) = \sigma^{i+1}$. So, $\sigma^{i+1} = STEP(\sigma^i, i+1) \downarrow$, $\sigma^{i+1} = STEP(\rho, i+1)$, $\sigma^i \in T_i$, $\rho \in T_i$, and $i+1 \leq q$. Therefore $\sigma^i = \rho$ by Lemma 7. From definition of STEP, $\sigma^{i+1} = \sigma^i \cup (q + B_{i+1})$ and $\sigma^i \cup (q + B_{i+1}) = \emptyset$. Hence $\sigma^i_p \subseteq \sigma^{i+1}_p$ and $\sigma^i_q = \sigma^{i+1}_q$.

- If $\sigma^{i+1} \in T^-_{i+1} = -(T_i, i+1)$, then $i+1 \notin \sigma^{i+1} \wedge \sigma^{i+1}_q = \sigma^i_q$. Therefore $i+1 \notin \sigma^i$. But $\sigma^i \in T_i$ by Lemma 2. As $STEP(\sigma^i, i+1) \downarrow$ and $i+1 \notin \sigma$, we deduce that $STEP(\sigma, i+1) = \sigma^i$. That is, $\sigma^{i+1} = \sigma^i$. Hence, in this case, we have $\sigma^i_p = \sigma^{i+1}_p$ and $\sigma^i_q = \sigma^{i+1}_q$.

□

Lemma 11. *For each i, k such that $1 \leq i \leq q$ and $0 \leq k < i$, and for every molecule $\sigma \in T_0$ such that $\sigma^i \downarrow$, we have $\sigma^k_q = \sigma^i_q$ and $\sigma^k_p \subseteq \sigma^i_p$.*

Proof. Let us prove the result by induction on i. The case $i = 1$ is trivial because there is no k such that $1 \leq k < i = 1$.

Let us assume the result holds for i such that $i \geq 1$ and $i < q$. Let k be such that $1 \leq k < i+1$. Let $\sigma \in T_0$ such that $\sigma^{i+1} \downarrow$.

If $k < i$ we have $\sigma^i \downarrow$, because $\sigma^{i+1} \downarrow$ and $\sigma^{i+1} = STEP(\sigma^i, i+1)$. From de induction hypothesis we deduce that $\sigma^k_q = \sigma^i_q$ and $\sigma^k_p \subseteq \sigma^i_p$. But $\sigma^i_q = \sigma^{i+1}_q$ and $\sigma^i_p \subseteq \sigma^{i+1}_p$ by Lemma 10. Therefore, $\sigma^k_q = \sigma^{i+1}_q \wedge \sigma^k_p \subseteq \sigma^{i+1}_p$.

The case $k = i$ directly follows from Lemma 10. □

4.1 Soundness of the Program

To establish the soundness of the program above designed, we consider the formula:

$$\theta(i) \equiv \forall \tau \in T_i \, \forall k, k' \in \tau \; (1 \leq k < k' \leq i \rightarrow B_k \cap B_{k'} = \emptyset)$$

That is, the formula $\theta(i)$ means that every molecule of tube T_i encodes a subfamily of $\mathcal{F}$ where the sets with indexes less or equal to i are pairwise disjoint.

Theorem 1. *The formula $\theta(i)$ is an invariant of the main loop. That is, for each i such that $1 \leq i \leq q$ we have the formula $\theta(i)$ is true.*

Proof. By induction on i. The case $i = 1$ is trivial.

Let i be such that $i \geq 1$ and $i < q$. Let us assume that the formula $\theta(i)$ is true. Let $\tau \in T_{i+1}$ and $k, k' \in \tau$ such that $1 \leq k < k' \leq i+1$. Since $\tau \in T_{i+1}$ and $i + 1 \leq q$, applying Lemma 9 there exists $\sigma \in T_0$ such that $\sigma^{i+1} = \tau$.

If $k' < i + 1$, since $k, k' \in \tau = \sigma^{i+1}$, $1 \leq k < k' \leq q$ and $\sigma_q^{i+1} = \sigma_q^i$, we obtain that $k, k' \in \sigma^i$. From $\sigma^{i+1} \downarrow$ and $\sigma^{i+1} = STEP(\sigma^i, i + 1)$, we have $\sigma^i \downarrow$. From Lemma 2, it can be deduced that $\sigma^i \in T_i$. Therefore $k, k' \in \sigma^i$, $1 \leq k < k' \leq i$, and $\sigma^i \in T_i$. As formula $\theta(i)$ is true by induction hypothesis, we conclude that $B_k \cap B_{k'} = \emptyset$.

If $k' = i+1$, we must prove that $B_k \cap B_{i+1} = \emptyset$. Since $k, k' \in \sigma^i$, from Lemma 2 we obtain $\sigma^i \in T_i$. Hence, $\sigma^i \in +(T_i, k') = +(T_i, i+1) = T_{i+1}^+$.

Now we can prove that $B_k \cap B_{i+1} = \emptyset$; that is, it does not exist any u such that $1 \leq u \leq r_{i+1} \wedge x_{i+1}^u \in B_k$.

- Let us suppose the opposite, that is, there exists u such that $1 \leq u \leq r_{i+1}$ and $x_{i+1}^u \in B_k$. Then, there would exist v such that $1 \leq v \leq r_k$ and $x_k^v = x_{i+1}^u$. In this case, we have $k \in \sigma^i$, $1 \leq k \leq q$, $\sigma_q^i = \sigma_q^{k-1}$ and $k \in \sigma^{k-1}$. Since $\sigma^k = STEP(\sigma^{k-1}, k)$ and $k \in \sigma^{k-1}$, from the definition of function $STEP$ we obtain that $\sigma^k = \sigma^{k-1} \cup (q + B_k)$ and $\sigma^{k-1} \cap (q + B_k) = \emptyset$. As $1 \leq v \leq r_k$ we have $(q + x_k^v) \in \sigma^k$. Since $k < k' = i + 1$, it holds $k \leq i$. Applying Lemma 11 we deduce that $\sigma_p^k \subseteq \sigma_p^i$. Therefore $(q + x_k^v) \in \sigma^i$, or, similarly, $(q + x_{i+1}^u) \in \sigma^i$. As $\sigma^{i+1} \downarrow$ and $i + 1 \in \sigma^i$, from the definition of function $STEP$ we conclude that $\sigma^i \cap (q + B_{i+1}) = \emptyset$. In particular, $(q + x_{i+1}^u) \notin \sigma^i$, which is impossible.

□

Corollary 1 (Soundness). *Every molecule of the output tube encodes a pairwise disjoint subfamily of $\mathcal{F}$.*

Proof. From Theorem 1 it follows that the formula $\theta(q)$ is true. Hence,

$$\forall \tau \in T_q \, \forall k, k' \in \tau \, (1 \leq k < k' \leq q \rightarrow B_k \cap B_{k'} = \emptyset)$$

□

4.2 Completeness of the Program

To establish the completeness of the designed program we consider the formula

$$\delta(i) \equiv \forall\, \sigma \in T_0\; ((\forall\, k, k' \in \sigma\; (1 \le k < k' \le q \rightarrow B_k \cap B_{k'} = \emptyset)) \rightarrow \sigma^i \in T_i)$$

That is, the formula $\delta(i)$ means that for every molecule of the input tube encoding a pairwise disjoint subfamily of $\mathcal{F}$, the molecule obtained from it after ith step will be in tube T_i.

Theorem 2. *The formula $\delta(i)$ is an invariant of main loop. That is, for each i such that $1 \le i \le q$ we have the formula $\delta(i)$ is true.*

Proof. By induction on i. Let $\sigma \in T_0$ be such that $\forall\, k, k' \in \sigma\; (1 \le k < k' \le q \rightarrow B_k \cap B_{k'} = \emptyset)$.

- If $1 \notin \sigma$, then $\sigma^1 \downarrow$. From Lemma 2 we deduce that $\sigma^1 \in T_1$.
- If $1 \in \sigma$, then $\sigma_p \equiv 0$. In particular, $\forall\, j\; (1 \le j \le r_1 \rightarrow \sigma(q + x_1^j) = 0)$. Hence $\sigma^1 \downarrow$. From Lemma 2 we obtain $\sigma^1 \in T_1$.

So, the result is true for $i = 1$.

Let us assume the result holds for i such that $i \ge 1$ and $i < q$. Let $\sigma \in T_0$ such that $\forall\, k, k' \in \sigma\; (1 \le k < k' \le q \rightarrow B_k \cap B_{k'} = \emptyset)$. From the induction hypothesis, we have $\sigma^i \in T_i$. If $i + 1 \notin \sigma^i$ then $STEP(\sigma^i, i + 1) = \sigma^i$. So, $\sigma^{i+1} \downarrow$. From Lemma 2 we deduce that $\sigma^{i+1} \in T_{i+1}$.

Let us suppose that $i + 1 \in \sigma^i$. From the definition of the function STEP and taking into account that $\sigma^i \downarrow$, it suffices to show that $\sigma^i \cap (q + B_{i+1}) = \emptyset$. On the contrary, there must be $j_0 = \min\{j : 1 \le j \le r_{i+1} \wedge (q + x_{i+1}^j) \in \sigma^i\}$. In this case we would have $(q + x_{i+1}^{j_0}) \in \sigma^{i-1}$.

- Indeed: if $(q + x_{i+1}^{j_0}) \notin \sigma^{i-1}$, then $\sigma^{i-1} \ne \sigma^i$. Taking into account that $\sigma^i = STEP(\sigma^{i-1}, i)$, we have

  $$i \in \sigma^{i-1} \wedge \sigma^i = \sigma^{i-1} \cup (q + B_i) \wedge \sigma^i \cap (q + B_i) = \emptyset$$

  So, there would be u such that $1 \le u \le r_i \wedge x_i^u = x_{i+1}^{j_0}$. Hence we would obtain $B_i \cap B_{i+1} \ne \emptyset$.

If $(q + x_{i+1}^{j_0}) \in \sigma^{i-1}$, then there exists $k_0 = \min\{k : (q + x_{i+1}^{j_0}) \in \sigma^k\}$. As tube T_0 is a $(p+q,q)$-library, we would have $k_0 > 0$ (because $\sigma_p \equiv 0$). Taking into account that $\sigma^{k_0} = STEP(\sigma^{k_0-1}, k_0)$, and $(q + x_{i+1}^{j_0}) \notin \sigma^{k_0-1}$, we would deduce that $\sigma^{k_0} \neq \sigma^{k_0-1}$. Hence, we have $k_0 \in \sigma^{k_0-1}$, $\sigma^{k_0} = \sigma^{k_0-1} \cup (q + B_{k_0})$ and $\sigma^{k_0-1} \cap (q + B_{k_0}) = \emptyset$. That is, there exist v such that $1 \leq v \leq r_{k_0}$ and $x_{i+1}^{j_0} = x_{k_0}^{v}$. Therefore, $B_{i+1} \cap B_{k_0} \neq \emptyset$, with $k_0 \neq i+1$. This contradicts our assumption. □

Corollary 2 (Completeness). *Every molecule of the input tube encoding a pairwise disjoint subfamily of $\mathcal{F}$ survives all the steps of main loop and will be in the output tube, T_q.*

Proof. From Theorem 2 it follows that the formula $\delta(q)$ is true. Hence,

$$\forall\, \sigma \in T_0 \; ((\forall\, k, k' \in \sigma \; (1 \leq k < k' \leq q \rightarrow B_k \cap B_{k'} = \emptyset)) \rightarrow \sigma^q \in T_q)$$

□

5 Applications

In this section we apply the above results to give efficient molecular solutions of two well-known numerical **NP**-complete problems, the *Exact Cover* problem and the *Set Packing* problem, within the *sticker model*.

The *Exact Cover* problem is the following: *Let $A = \{1, \ldots, p\}$ a finite set, and let $\mathcal{F} = \{B_1, \ldots, B_q\}$ a finite family of subsets of A. Determine whether there exist $i_1, \ldots, i_k$ such that $\{B_{i_1}, \ldots, B_{i_k}\}$ is a partition of A.*

A molecular program within the *sticker model* solving the *Exact Cover* problem is the following one:

```
Procedure Exact_Cover
Input:  $T_0$ (a $(p+q,q)$-library)
        $T_0 \leftarrow$ Disjoint($T_0$)
        for $i=1$ to $p$ do
            $T_0 \leftarrow +(T_0, q+i)$
        end for
```

The number of molecular operations of this program is of the order of $O(q \cdot p)$. The number of used tubes is of the order of $O(p)$. The number of molecules of the input tube is of the order of 2^q.

The soundness and the completeness of this molecular program directly follows from the formal verification of the `Disjoint` procedure.

The *Set Packing* problem is the following: *Let $A = \{1, \dots, p\}$ a finite set, let $\mathcal{F} = \{B_1, \dots, B_q\}$ a finite family of subsets of A, and let $k \in \mathbb{N}$. Determine whether a pairwise disjoint subfamily of $\mathcal{F}$ with cardinal k exists.*

A molecular program within the sticker model solving the *Set Packing* problem is the following one:

```
Procedure Set_Packing
Input:  $(T_0,\ k)$ (where $T_0$ is a $(p+q,q)$-library)
        $T_1 \leftarrow$ Disjoint$(T_0)$
        $T_{out} \leftarrow$ Cardinal_Sort$(T_1, 1, q)[k]$
        Read$(T_{out})$
```

In this program the `Cardinal_Sort` subroutine is used, which allows us to sort, according to their cardinality, the elements of the family encoded by T_1 (see [2] for further details).

The number of molecular operations of this program is of the order of $O(q{\cdot}p{+}q^2)$. The number of used tubes is of the order of $O(\max\{p, q\})$. The number of molecules of the input tube is of the order of 2^q.

The soundness and the completeness of this molecular program directly follows from the formal verification of the `Disjoint` and the `Cardinal_Sort` procedures (see [2] for further details).

6 Conclusions

In this paper, a molecular program generating pairwise disjoint families has been presented within the sticker model. Using this program as a subroutine we have given molecular solutions of two well-known numerical **NP**-complete problems: the *Exact Cover* problem and the *Set Packing* problem. Moreover, we present a detailed study of the formal verification process of a program within a molecular model with memory, proving the *soundness* and *completeness* of the designed program. For that, invariant formulas (in the variables that appear in the program) of the main loop have been searched through the semantics of the program and the problem, using inductive techniques.

The study of the formal aspects of molecular programs is a necessary first step for the automatic processing of them by means of reasoning systems (we are currently working on ACL2 and PVS). The production of prototypes which are able to be executed regarding to molecular computational models within the framework of theorem provers, will allow to automate both, the soundness and completeness, of molecular programs.

Acknowledgement

The support for this research through the project TIC2002-04220-C03-01 of the Ministerio de Ciencia y Tecnología of Spain, cofinanced by FEDER funds, is gratefully acknowledged.

References

[1] Garey, M. R.; Johnson, D. S. *Computers and intractability*, W. H. Freeman and Company, New York, 1979.

[2] Pérez-Jiménez, M. J.; Sancho-Caparrini, F. Solving Knapsack Problems in a Sticker Based Model, in Jonoska, N.; Seeman, N. (eds.), *DNA Computing*, LNCS **2340** (2002), 161-171.

[3] Roweis, S.; Winfree, E.; Burgoyne, R.; Chelyapov, N.; Goodman, M; Rothemund, P; Adleman, L. A Sticker–Based Model for DNA Computation, *Journal of Computational Biology*, **5**, 4 (1998), 615–629.