ROMANIAN JOURNAL OF INFORMATION SCIENCE AND TECHNOLOGY
Volume **12**, Number 2, 2009, 139–155

Cooperative construction of pointed pictures

Paolo BOTTONI[1], Anna LABELLA[1]

[1] Department of Computer Science, University of Rome *Sapienza*
Via Salaria 113, Roma, Italy, 00198
E-mail: {bottoni,labella}@di.uniroma1.it

Abstract. We explore a variant of a recently introduced operation on images, which provides an adequate basis for modeling computations in which concurrent agents cooperatively construct (pointed) pictures. In this setting, concurrent agents generate languages of multi-dimensional words on partially ordered alphabets through a simple operation of *overlapping*, constrained by the order imposed on the alphabet. The overlapping operation is proved to be a powerful tool for picture generation in general. The operation is parametric with respect to the composition law, and we show how some simple requests on the behavior of this law provide a meet-semilattice structure to the class of pointed pictures. This feature allows their use in building a model for concurrent processes in the spirit of process algebras.

1. Introduction

Formal models of 2-dimensional languages usually suffer by the lack of an adequate algebraic framework in which to set the typical matching and replacing processes involved in rewriting. Such processes are suitably represented in one-dimensional rewriting due to the monoid structure deriving from the properties of the concatenation operation, or in high-level replacement approaches exploiting pushouts in (weak) adhesive high-level categories [16] and the gluing operation. In general, these models implicitly refer to some underlying spatial structure providing support to the execution of computations. Hence, Chomsky-like grammars work on linear strings, term rewriting exploits trees, parallel models, such as cellular automata, work on regular meshes, and distributed computing on some type of network. For each of these

models, adequate algebraic or categorical settings are available. A particular deficiency under this respect seems, on the contrary, to plague 2-dimensional rewriting. In particular, vertical and horizontal versions of traditional 1-dimensional concatenation provide a simple analogous for grammar-style rewriting, but their being partial functions makes the consequent algebraic structure quite poor. On the other hand, most rewriting processes implicitly rely on some notion of superposition, which is not sufficiently supported by 2-dimensional forms of concatenation.

In [6] two new operations, of shifting and superposition, were shown to provide an adequate account of several aspects of 2-dimensional rewriting. In [2], we proposed to equip pictures with some additional information about their availability for subsequent applications of the operation. This allowed the definition of a single operation on pairs of *pointed pictures*, thereby imposing a monoid structure on them. Moreover, the resulting structure can be uniformly extended to any type of n-dimensional structure.

In this paper, we exploit a simpler version of pointed pictures, which reduces the amount of additional information, but is still sufficient to model the important phenomenon of cooperative construction of pictures by different agents. In this setting, a picture is seen as the space on which different agents may contribute to its final appearance. Agents can work concurrently on different parts of the image, without being forced to act sequentially on contiguous zones, or take turns according to some policy. However, overlapping contributions can also be managed according to some partial ordering imposed on the alphabets from which the pictures are formed. Moreover, we provide some axioms required for the definition of the overlapping operations, through which we can parameterize operations according to the law used for composing contributions from different agents. A nice algebraic structure still results, namely that of meet-semilattice.

After analyzing related work on pictures in Section 2, Section 3 motivates the approach through some examples of concurrent computation on pictures. Section 4 elaborates on overlapping of pointed (multi-dimensional) words, while Section 5 discusses the power of the combination of orders on alphabets, overlapping and filtering conditions, by simulating several language definition devices for one-dimensional and double stranded strings, and for pictures. Finally, Section 6 concludes the paper and points to topics for future work.

2. Related work

The exploration of the algebraic characterization of 2-D sentences and languages started with Kirsch [14] and Dacey [7], imposing some form of juxtaposition (which in the 1-D case reduces to common concatenation) on the bidimensional structure of images. Kirsch proposes elementary patterns in which specific symbols indicate the role they will play in a figure obtained through a bidimensional grammar. Rule application consists of the superposition of the right-hand side pattern on a context in which the left-hand side pattern occurs. Kirsch shows how to generate triangles, while Dacey generalizes the approach to several figures and shows a group structure for languages describing polygons related by rotations.

The use of a notion of superposition as an equivalent to juxtaposition originated a number of picture rewriting systems *e.g.* array grammars, with different types of control [17].

In another direction, horizontal and vertical versions of concatenation were proposed in [21], giving rise to typical forms of rewriting in which a regular grammar is used to generate a starting horizontal word, from which a context free grammar starts producing columns [20]. The problem with these versions of concatenation is that they are partial functions, being applicable only to pictures of compatible sizes, along the horizontal or the vertical axis. An algebraic characterization of pictures based on horizontal and vertical concatenation has been given in the form of doubly ranked monoids constructed from alphabets of bidimensional elements, in which each individual operation is associative and compatible with each other [9]. Diagonal versions have also been studied to allow concatenation of pictures of triangular, rather than rectangular, shape [10]. A survey on these topics, with particular emphasis on the notion of recognisability of picture languages, is in [8].

The proposal of pointed pictures in [2] stems from the limitations of a previous proposal based on a pair of operations, called *shift* and *sup*, which exploits the introduction of the transparent symbol. Their properties, in particular with respect to their generating power, were studied in [6].

Pointed pictures exploit an analogous of attachment positions for contour curves, introduced by Shaw to allow a form of curve composition where the head of an element is attached to the tail of another [19]. In particular, [15] introduces pointed drawings, described by a string of directions and a pair of positions, called the departure and arrival points of the drawing. The concatenation (superposition) of two drawings is possible only if the position of the arrival point of the first coincides with that of the starting point of the second. Figures are classes of equivalence of drawings modulo translation, and concatenation is obtained by taking a pair of translated versions such that the coincidence condition on arrival and departure points is satisfied. For pointed figures, concatenation is a total and associative function. The resulting algebraic structure is a finitely generated inverse monoid, the set of generators being coloured pixels with all possible positions for the arrival and departure points [15].

An extensive study on the (im)possibility of recovering concatenation of pictures, as distinct from figures, in 2-D by simply enriching them with attachment points, or by allowing placement only over well-behaved paths, is in [4]. The proposal in [2] avoids such limitations, as pointed pictures bring with them information on attachment points and picture orientation, but allows their translation and rotation.

3. Collaborative construction of images

The definition of a suitable setting for superposition operations is increasingly important, as collaborative construction of spatially-related information is attracting attention. Applications such as GoogleEarth or GoogleMap provide a widely exploited support for the addition of proprietary or public information. In a completely different arena, real or virtual whiteboards are increasingly used in collaborative creative

settings, to exchange ideas, illustrate variants of a solution, annotate designs. Leisure applications are also gaining diffusion, such as Microsoft Surface, in which an active surface acts as a medium for the presentation and interactive exchange of pictures.

In all these cases, problems arise as to which information should be presented on the visible surface of the application, how can a participant get control of an area, how to resolve possible conflicts in case of overlapping contrasting information. We illustrate these problems through some motivating examples, abstracting the surface on which the collaborative information can be constructed as a picture, a function $\pi : R \times C \to \Sigma$, where R and C are initial segments of the natural numbers, and Σ is a finite alphabet of symbols, including a special symbol τ, called the *transparent* symbol. Moreover, we assume the existence of a partial order $<$ in Σ, such that $\forall x \in \Sigma \setminus \{\tau\}$, $\tau < x$. The ordering is used to solve possible conflicts in the colouring of a pixel, so that a position $(r, c) \in R \times C$ coloured with a symbol x can change its colour to a symbol y only if $x < y$.

We present a cooperative version of the 4-colour problem, i.e. finding a colouring of a map with $K > 4$ regions, using up to four colours, so that no two adjacent regions are coloured in the same way. We model this as a game in which 4 agents, each endowed with a colour and with a private set of pictures, take turns in colouring a different region of a shared picture at each time. Colouring a region is modelled as overlapping one of the pictures available to the agent to the current state of the game, representing the move for the agent at that turn. Hence, we consider $\Sigma = \{c_i \mid i = 1, \ldots, 4, \} \cup \{\tau\}$, where τ indicates a pixel in the shared picture which has not been coloured yet, or a pixel in a private picture of an agent that the agent is not entitled to colour. We assume a flat ordering on Σ, i.e. $x \neq \tau \Rightarrow \nexists y \in \Sigma, x < y$, meaning that no agent can colour a pixel already coloured in the shared picture. Initially, the shared image Π has all pixels transparent. Each agent a_i, $i = 1, \ldots, 4$ possesses a set of pictures $\Pi_i = \{\pi_i^1, \ldots, \pi_i^K\}$, with the same size as Π. Each $\pi_i^j \in \Pi_i$ has all pixels transparent, except for region j, which is coloured with $c_i \in \Sigma$. If a complete colouring of Π is reached, i.e. Π has no transparent pixel, the winner is the agent which coloured the overall maximum surface; otherwise, there is no winner. In a variant of the game the winning condition could refer to the number of regions an agent has contributed. In an iterated version of the game, there might be penalties for not completing the picture and prizes to every agent, proportional to the contribution of the agent to the completed picture.

Figure 1 shows an initial configuration for this game, with names assigned to the agents. In Figures 1–3, we have separated regions by lines, which are not part of the original map.

Agents can take turns in colouring the image, or they can operate concurrently, possibly conflicting on a same region, in which case the conflict is resolved (by accepting only one contribution) according to some policy unknown to the agents. Agents can follow different strategies, for example, non-deterministically choosing a region as the next one for their colouring, or choosing to colour each time the region with the minimal number of neighbouring region, or with the maximal area. In any case, due to the flat ordering, the colour of a region cannot be changed once an agent has placed its contribution.

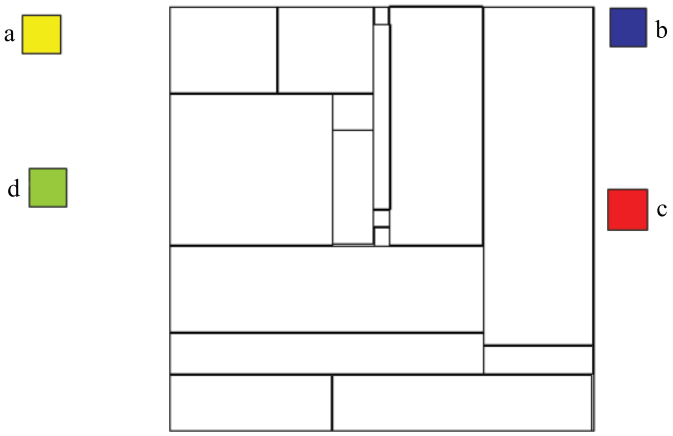

Fig. 1. An initial configuration for the 4-colour game.

Figure 2(a) shows a configuration after 8 turns, and Figure 2(b) a final configuration reachable from the first, in the case that the alphabetic order determines agents' turns, and each agent chooses non deterministically the next area to color.

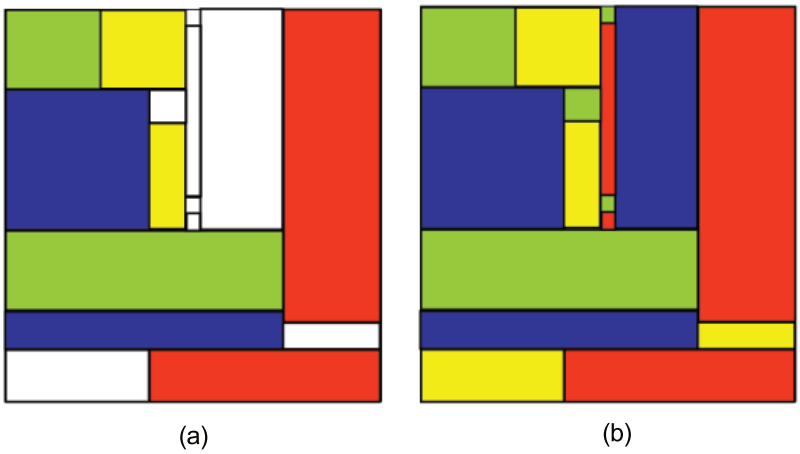

Fig. 2. A possible state of the game after 8 moves (a), and a final configuration (b), when all agents follow a non deterministic strategy.

Figure 3(a) shows a configuration, after 14 moves, from which the game cannot be completed, in a different instantiation of the game for the same initial map and the same colours. Here, each agent follows the strategy of selecting each time among the available regions with the minimal number of neighbours. The numbers in the area represent the number of neighbours for each area. The same strategy achieves success for the simpler map of Figure 3(b).

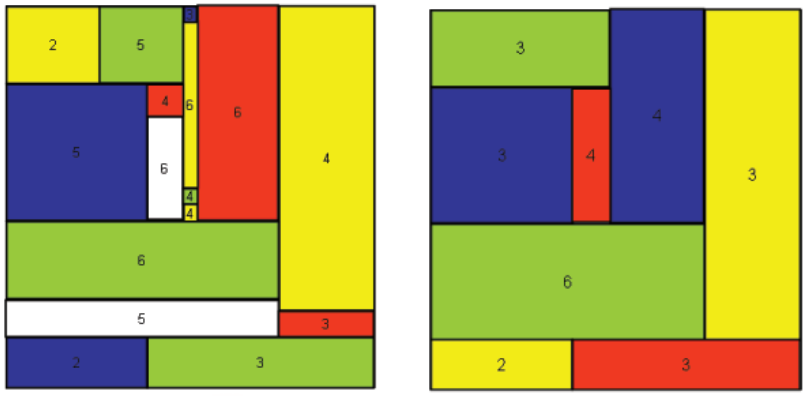

Fig. 3. A configuration from which the game cannot be completed (a), and a final configuration (b), for a strategy choosing regions with fewer neighbours.

The second case we consider is one in which different travel agents cooperate to label pixels corresponding to hotels in a thematic map with labels composed of four-tuples over the alphabet $\Sigma = \{(\{l, m, h\} \times [0..1])^4\} \cup \{\tau\}$. The letters stand for *low*, *medium*, *high* and the tuples are formed with judgements on the quality of *service*, *view*, *comfort*, *food*. τ indicates the absence of judgement, and the ordering on Σ is coherent with ordering on judgement components, i.e. $((x_1, j_1), (x_2, j_2), (x_3, j_3), (x_4, j_4)) \leq ((y_1, i_1), (y_2, i_2), (y_3, i_3), (y_4, i_4))$ if and only if $j_k \leq i_k$ for $k = 1, \ldots, 4$. Different agents can be more qualified in different sectors, so that the second component of each pair in the label indicates the level of qualification for each argument. Each agent receives a copy of some region of the initial picture, with an indication of the positions to fill. These copies may be different, in that agents may not be requested to visit all hotels, but consider only some of them. Agents can thus operate independently of one another. The construction of the final picture is managed by a central agency, receiving the judgements of the individual agents in the form of copies of the original regions annotated with judgements. Figure 4 shows the independent judgments, and their composition, of agents A and B visiting different, non disjoint, regions of the map. Instead of repeating the qualification for each position, we have indicated the levels for each agent. Two hotels have been judged by both agents, who disagree on the quality of service for the hotel located at $(2, 6)$ and on the quality of comfort for that at $(4, 4)$. In the composition one considers that A is more qualified for judging service and B for comfort.

Finally, the case of two agents remotely cooperating through whiteboards can be modelled in the following way. Each agent can directly manipulate its whiteboard, and transmit parts of it to the collaborating agent, possibly letting available for modifications some of the figures it produced. In this case one defines $\Sigma = \Sigma_0 \cup \Sigma_0' \cup \{\tau\}$, with $\Sigma_0' = \{\alpha' \mid \alpha \in \Sigma_0\}$, with an ordering defined by the following rule: $(x, y) \in < \iff (x = \tau) \vee (x \in \Sigma_0 \wedge y \in \Sigma_0')$, so that a primed colour is one which

cannot be changed. In this case, a same zone can be modified indefinitely until an agent places a lock (i.e. a primed version of the colour) on it.

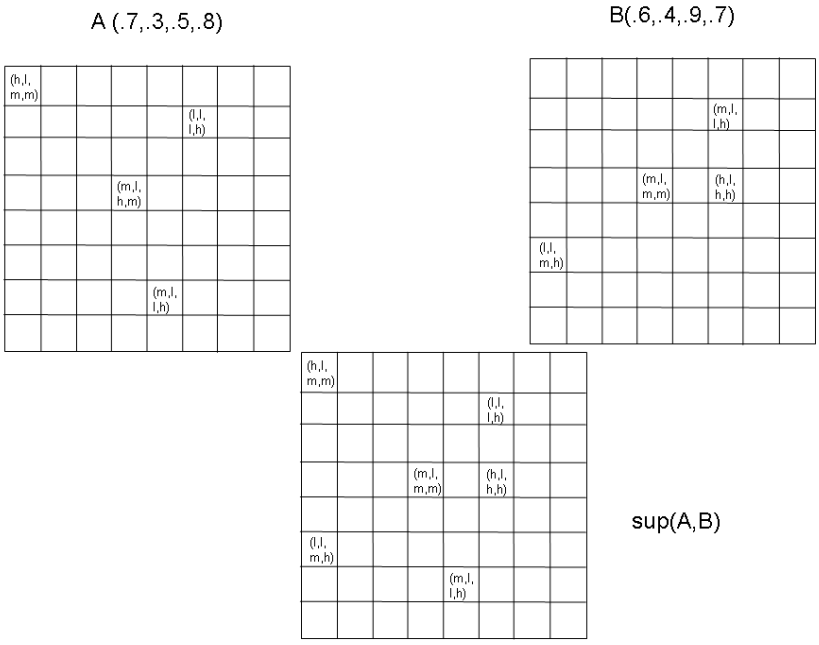

Fig. 4. The two independent judgements of the two agents and their composition.

4. Overlapping of pointed pictures

We now provide a formal model of computing with pictures, based on the existence of partially ordered alphabets, restricting the notion of pointed word proposed in [2], so that orientation (and consequently rotation) is not considered. This model accommodates the intuition of cooperative and concurrent construction of pictures discussed in Section 3, which supports the introduction of a uniform monoidal structure on words of any dimension.

We first recall the definition of meet-semilattice.

Definition 1 (meet-semilattice). A *complete meet-semilattice* $\mathbf{L} = (L, \leq, \wedge, \bot)$ is a partial order with greatest lower bound for any non-empty family of elements, $\wedge$. The bottom element is denoted by $\bot$.

A *complete meet-semilattice monoid* is a monoid $(L, \bullet, 1)$ such that the prefix relation between its elements (as usual defined as $s \leq t$ iff there exists $u \in L$ such that $s \bullet u = t$) is an order and induces a complete meet-semilattice structure.

Let Σ be a partially ordered finite alphabet, structured as a complete meet-semilattice with bottom element denoted by τ. Σ can be used to build a meet-semilattice of pictures Π with monoidal structure[1]. For the sake of understandability, we start with 1-dimensional pictures, where the monoidal structure can be thought of as a generalization of strings concatenation.

Definition 2 (Pointed string).

1. A *string* w on Σ is a function $w : Z \to \Sigma$, where Z is the set of integer numbers and w is almost everywhere equal to τ.
2. A *pointed string* (w, x_e, x_t) is a string w with two designated positions x_e and x_t, called *entry* and *exit* respectively, which can also coincide.
3. A *translation* t_k of (w, x_e, x_t) by k positions is a pointed string $t_k((w, x_e, x_t)) = (w', x'_e, x'_t)$, where $w'(x) = w(x - k)$, $x'_e = x_e - k$, $x'_t = x_t - k$.

Non-transparent symbols need not be contiguous in w. This reminds of partial words, where holes may appear in finite strings [1]. Rather than indicating any symbol in the current string, τ indicates a position which can be occupied by any symbol via an overlapping operation. In the rest of the paper the set of pointed strings is considered modulo equivalence with respect to translations. This means that we will use translations in the sequel, for both comparing and composing pointed strings. The set of pointed strings PS_Σ (considered up to translations) is partially ordered according to Definition 3. We use PS when Σ is understood.

Definition 3 (Partial order on pointed strings). $(w_1, x_{11}, x_{21}) < (w_2, x_{12}, x_{22})$ if there exists a translation by $x_{12}-x_{11}$ such that $(w'_2, x'_{12}, x'_{22}) = t_{x_{12}-x_{11}}((w_2, x_{12}, x_{22}))$, $w_1 < w'_2$ as functions (i.e. $w_1(x) < w'_2(x), \forall x \in Z$) and $x'_{12} = x_{11}, x_{21} \leq x'_{22}$. (Note that only strings where $x_{11} - x_{21} \leq x_{12} - x_{22}$ are comparable.)

Proposition 1. *Let* $\mathbf{w_0} = (w_0, 0, 0)$*, where* w_0 *is the string with value* τ *everywhere. Then* $(PS, \leq, \wedge, \mathbf{w_0})$ *is a meet-semilattice, with* $\wedge$ *denoting the* meet *operation.*

We now define a family of overlapping operations $\bullet_\&$ on PS, parametrically w.r.t. a binary associative operation $\& : \Sigma \times \Sigma \to \Sigma$ as follows:

$$(w_1, x_{11}, x_{21}) \bullet_\& (w_2, x_{12}, x_{22}) = (w, x_{11}, x'_{22})$$

with $w(x) = w_1(x)$ & $w'_2(x)$, where $\bullet_\& : PS \times PS \to PS$ is canonically defined by the pointwise application of &, and w'_2 is the translation of w_2 by $x_{12} - x_{21}$.

Figure 5 shows three different instantiations of the overlapping operation for two pointed strings σ_1 and σ_2. From left to right, we show the application of:

1. an operation $\&_1$ which maintains the value of the first string in positions where both strings are non-transparent;

[1] Actually Π can accommodate several monoidal structures.

2. an operation $\&_2$ which preserves the second string;

3. an operation $\&_3$ which combines non transparent colours from the two strings.

The transparent positions are left understood. The characters inside the pixels indicate the 0, entry and exit positions for each string. Note that, irrespective of the adopted operation, all the resulting strings present the same designated positions.

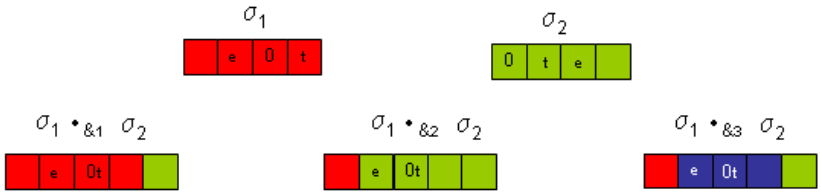

Fig. 5. Instantiation of overlapping between pointed strings.

Proposition 2. *Let* $\& : \Sigma \times \Sigma \to \Sigma$ *be an associative monotonic operation with* τ *as unit, satisfying the following:*
$\forall \sigma_1, \sigma, \sigma' \in \Sigma$, *:*

- $\sigma \leq \sigma'$ *implies* $\sigma_1 \ \& \ \sigma \leq \sigma_1 \ \& \ \sigma'$ ***(right monotonicity)***
- $\sigma_1 \ \& \ (\sigma \wedge \sigma') = \sigma_1 \wedge \sigma \ \& \ \sigma_1 \wedge \sigma'$ ***(right semidistributivity)***
- $\sigma_1 \leq \sigma_1 \ \& \ \sigma$ ***(non decreasing property)***

Then $(PS, \leq, \bullet_\&, \wedge, \mathbf{w_0})$ *with the overlapping operation* $\bullet_\&$ *defined pointwise and coordinatewise as above, is a meet-semilattice with a monoidal structure.*

Remark 1. *In the examples of Figure 5,* $\&_1$ *satisfies the conditions of Proposition 2, while* $\&_2$ *does not. For* $\&_3$*, the conditions must be required for the relation between the individual colours and their combinations.*

The traditional operations of *concatenation* and *anticoncatenation* can now be recovered by suitably fixing the entry and exit points in the strings.

Definition 4 extends these notions to the case of 2 (or more)-dimensional pictures. In this case, positions are defined by vectors of coordinates, denoted $\overrightarrow{x}$. For simplicity, the graphical examples will exploit one- or bi-dimensional structures only.

Definition 4 (Pointed picture).

1. A *picture* π on Σ is a function $\pi : Z^n \to \Sigma$, where Z is the set of integer numbers and π is almost everywhere equal to τ.

2. A *pointed picture* $(\pi, \overrightarrow{x}_e, \overrightarrow{x}_t)$ is a picture with two designated *entry* and *exit* positions.

3. A *translation* of a pointed picture $(\pi, \overrightarrow{x}_e, \overrightarrow{x}_t)$ by $\overrightarrow{k}$, also noted $t_{\overrightarrow{k}}$, is a pointed picture $(\pi', \overrightarrow{x}'_e, \overrightarrow{x}'_t)$, where $\pi'(\overrightarrow{x}) = \pi(\overrightarrow{x} - \overrightarrow{k})$, $\overrightarrow{x}'_e = \overrightarrow{x}_e - \overrightarrow{k}$, $\overrightarrow{x}'_t = \overrightarrow{x}_t - \overrightarrow{k}$.
4. PP_Σ is the set of pointed pictures on Σ up to translations.

Definition 5 generalises to multidimensional pictures in the set PP_Σ the notion of partial order introduced in Definition 3. From now on, we use PP when Σ can be left understood.

Definition 5 (Partial order on pointed pictures). $(\pi, \overrightarrow{x}_e, \overrightarrow{x}_t) < (\pi', \overrightarrow{x}'_e, \overrightarrow{x}'_t)$ if and only if there exists a translation $(\pi'', \overrightarrow{x}''_e, \overrightarrow{x}''_t)$ of $(\pi', \overrightarrow{x}'_e, \overrightarrow{x}'_t)$ such that $\pi < \pi''$ as functions and $\overrightarrow{x}_e = \overrightarrow{x}''_e$, $\overrightarrow{x}_t \leq \overrightarrow{x}''_t$, i.e. $(\pi'', \overrightarrow{x}''_e, \overrightarrow{x}''_t) = t_{(\overrightarrow{x}'_e - \overrightarrow{x}_e)}((\pi', \overrightarrow{x}'_e, \overrightarrow{x}'_t))$.

In Definition 5, vectors are compared componentwise, i.e. $\overrightarrow{x} \leq \overrightarrow{y} \Leftrightarrow \forall i \overrightarrow{x}[i] \leq \overrightarrow{y}[i]$.

Proposition 3. *Let $\pi_{\mathbf{0}} = (\pi_0, \vec{0}, \vec{0})$, where π_0 is the picture with all pixels transparent. Then, $(PP, \leq, \wedge, \pi_{\mathbf{0}})$ is a meet-semilattice.*

As before, we define a family of overlapping operations $\bullet_{\&}$ on PP, parametrically w.r.t. a binary associative operation $\& : \Sigma \times \Sigma \to \Sigma$ as follows:

$$(\pi, \overrightarrow{x}_e, \overrightarrow{x}_t) \bullet_{\&} (\pi', \overrightarrow{x}'_e, \overrightarrow{x}'_t) = (\pi'', \overrightarrow{x}_e, \overrightarrow{x}''_t)$$

with $\pi''(\overrightarrow{x}) = \pi(\overrightarrow{x}) \bullet_{\&} \pi'''(\overrightarrow{x})$, where $(\pi''', \overrightarrow{x}'''_e, \overrightarrow{x}'''_t) = t_{(\overrightarrow{x}_t - \overrightarrow{x}'_e)}((\pi', \overrightarrow{x}'_e, \overrightarrow{x}'_t))$.

Figure 6 illustrates three instantiations of the overlapping operation for two pointed pictures, according to the same conventions as for Figure 5.

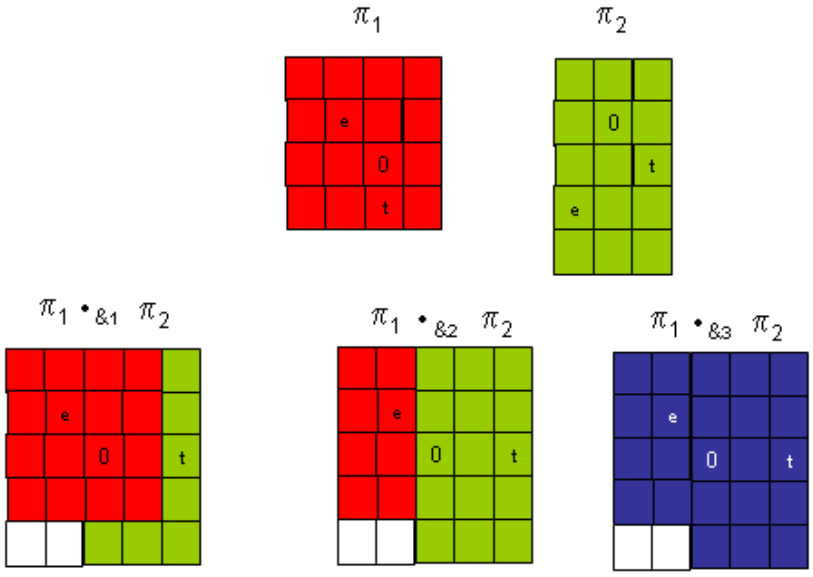

Fig. 6. Instantiations of overlapping between pointed pictures.

5. On the generative power of overlapping

Overlapping of pointed structures supports a generative definition of languages, which differs from traditional, concatenation-based rewriting or parsing mechanisms for strings, but which also provides a setting for languages of multidimensional words, more comprehensive than plain extensions of concatenation to several dimensions, as in [8].

A different approach to dealing with multidimensional words, based on the use of transparent symbols and the possibility of shift and superposition, has already produced several computability results. Moreover, DNA computing with its notion of complementarity, also can be characterized in terms of some & operation, as introduced in Section 4.

We will therefore first revise these notions, and then introduce an original model for rewriting exploiting the designated entry and exit positions of pointed pictures.

In [8], the notion of recognizable language is proposed, based on the equivalence of several formal devices for accepting languages of two dimensional words, on alphabets which do not contain the transparent symbol τ, but which present the special symbol #. In particular, we show how languages recognized by tiling systems can be generated with the use of the overlapping operation. To this end we exploit the notion of *adult language*, i.e. the subset of $L(\Pi_0)$ formed by words which can only rewrite into themselves. We indicate the adult language originated from Π_0 as $AL(\Pi_0)$. We also use a function *cut*, restricting a pointed word to the minimal word (i.e. forgetting designated points) containing all non transparent symbols. We first recall the notion of tiling system from [8].

Definition 6 (Tiling system). A *tiling system* is a 4-tuple $TS = (\Sigma, \Gamma, \Theta, \rho)$ with Σ and Γ alphabets, Θ finite set of tiles (blocks of size 2×2 on $\Gamma \cup \{\#\}$) and $\rho : \Gamma \to \Sigma$ a morphism.

The language recognized by TS is the set of pictures $L(TS) = \rho(L(\Theta))$ where $L(\Theta)$ is the set of all pictures which present only the allowed blocks in Θ. The set of all languages recognised by tiling systems is called $\mathcal{L}(\mathcal{TS})$.

Proposition 4. *Let $\phi_{\sim\blacktriangle}$ be the predicate which admits only pictures without the symbol $\blacktriangle$. Given a tiling system $TS = (\Sigma, \Gamma, \Theta, \rho)$, there exists a set Π_0 of pointed pictures on Γ and an operation $\&$ on $\Gamma \cup \{\blacktriangle\}$, s.t. $L(TS) = cut(\rho(AL(\Pi_0, \bullet_\&) \cap \Phi_{\sim\blacktriangle}))$.*

Proof. We place a partial order on $\Gamma \cup \{\blacktriangle\}$ as follows: $\forall x \in \Gamma, \tau < x < \blacktriangle$, while all elements of Γ are incomparable with one another. We define Π_0 as follows: $\Pi_0 = \{(\theta, (e_1, e_2), (x_1, x_2)) \mid \theta \in \Theta, e_i \in \{1, 2\}, x_i \in \{1, 2\}$ for $i = 1, 2$, if θ does not contain $\#$, e_i and x_i on positions not occupied by # otherwise$\}$. For $\& : \Gamma \cup \{\blacktriangle\} \times \Gamma \cup \{\blacktriangle\} \to \Gamma \cup \{\blacktriangle\}$ we choose the following definition:
1) $\tau \& x = x$
2) $x \& x = x$
3) $y \& z = \blacktriangle$
4) $\blacktriangle \& \blacktriangle = \blacktriangle$
for $x \in \Gamma$, $y, z \in \Gamma \cup \{\blacktriangle\}$, $y \neq z$.

Note that pictures in Π_0 do not contain τ in the positions (z_1, z_2), $1 \leq z_1 \leq 2$, $1 \leq z_2 \leq 2$. By considering all possible entry and exit positions, it is possible to generate every path in a picture without changing the value of its positions. To complete the proof, one can observe that $L(\Pi_0, \bullet_{\&}) \cap \Phi_{\sim\blacktriangle}$ only contains pictures obtained by overlapping elements in Π_0 in such a way that the value of a position either changes from τ to some $x \in \Gamma \setminus \{\tau\}$, or remains the same. The adult language obtained in this way contains all pictures where # form a rectangle filled with non-transparent symbols. Hence, $AL(\Pi_0, \bullet_{\&}) \cap \Phi_{\sim\blacktriangle} = L(\Theta)$. □

The notion of overlapping has an immediate reference to the notion of layer in [5].

Definition 7 (Superposition of layered strings). Let $x, y \in (\Sigma \cup \{\tau\})^*$. The superposition $x \diamond y$, is defined as follows. Let $K = max(\mid x \mid, \mid y \mid)$. Let $w' = w \bullet \tau^{K-|w|}$ for $w = x, y$. Then $z = x \diamond y$, with $\mid z \mid = K$, and $z(i) = x'(i)$, if $x'(i) \neq \tau$, $y'(i)$, otherwise.

Proposition 5. *Superposition of layered strings can be simulated by superposition of strings.*

Proof. The $\diamond$ operation coincides with the overlapping operation $\bullet_{\&_1}$, of the two pointed strings $(x, 0, 0)$ and $(y, 0, 0)$, according to the function $\&_1$ from the example in Figure 5. □

In [5] superposition is investigated as an abstract operation on languages, in particular studying properties of closure of families of languages in the Chomsky hierarchy with respect to it. The families REG, CS, RE are closed under superposition, and the families LIN, CF are closed under superposition with regular languages. These same results hold for $\bullet_{\&_1}$ after applying the morphism $point_0 : \Sigma^* \to PS(\Sigma)$, mapping each string in Σ^* into a pointed string with entry and exit position in the origin[2].

Sticker systems [11] allow a double stranded DNA string, with single stranded, *sticky* ends, to be prolonged by another if they have complementary portions of their single stranded ends. To achieve this, complementarity between symbols of the DNA alphabet is modelled through an *involution* operation. In particular, an *involution* over a set S is a bijective mapping $\bar{\cdot} : S \longrightarrow S$ such that $a = \overline{\overline{a}}$. If $\overline{a} = b$, then we say that a and b are *complementary*. The DNA alphabet $V_{DNA} = \{A, C, G, T\}$ is such that $\overline{A} = T$, $\overline{C} = G$. We also introduce an *amplification* operation, analogous to the polymerase chain reaction of DNA biochemistry, which allows the arbitrary replication of strings. In our case, amplification produces versions of a string with all possible non transparent positions for entry and exit.

Proposition 6. *The behaviour of sticker systems can be simulated exploiting overlapping, filtering and amplification.*

Proof. Consider an operation & defined by

1) $x \& \overline{x} = \top_{(x,\overline{x})}$
2) $x \& y = \blacktriangle$

[2]Actually, due to the equivalence relation induced by translation, any morphism $point_z$ will do for any $z \in Z$.

for $y \neq \overline{x}$, on an alphabet $\Sigma \cup \{\top_{(w,z)} \mid w, z \in \Sigma\} \cup \{\blacktriangle\}$.

We then introduce an ordering $<$ such that $\tau < x < \blacktriangle$, $x < \top_{(y,z)}$ for all $x, y, z \in \Sigma$ and any $x, y \in \Sigma$ are not comparable. The predicate $\phi_{\forall\top}$ selects only strings where all non transparent symbols are of type $\top_{(x,y)}$, and the predicate $\phi_{\sim\blacktriangle}$ is defined as above. By iterating overlapping, application of $\phi_{\sim\blacktriangle}$, and amplification, and finally applying $\phi_{\forall\top}$ to select only strings corresponding to complete double strands, one obtains an encoding of all the double stranded strings produced by a sticker system. □

In [11] it is shown that sticker systems generate all regular languages through weak coding of the produced strings. Moreover, if control words are used to define the order in which strings are attached to one another, and a demand is made on equality of the control words used to make the upper and the lower strand grow, one obtains a weak coding of recursively enumerable language. The distinction between upper and lower strand can be simulated by considering whether a string from Π_0 is used as first or second argument of the overlapping. Finally, for the case when the control words can be different, but have the same length, one obtains, through weak coding, a family of languages which is strictly comprised between *REG* and *RE*. Again, the control word mechanism can be applied to the simulation process based on overlapping of pointed strings.

We can observe that not all forms of bio-inspired operations can be simulated in this way. In particular, we observe that an analogous of the superposition operation from [5], extended in [3] to define Watson-Crick superposition, cannot be simulated with an overlapping operation conforming to the conditions of Proposition 2. In [3], *Watson-Crick morphisms* over the alphabet V are introduced as involution over V^*.

Definition 8 (Watson-Crick superposition). Let $x, y \in \Sigma^+$ be two non-null strings on Σ. The *Watson-Crick superposition* operation, denoted by $\diamond_{WK}$, is defined as follows: $x \diamond_{WK} y = \{z \in \Sigma^+ \mid$ one of the following conditions is satisfied:

1. $\exists i$ such that $(|x| - i + 1 \leq |y|)$ AND
   $(x[i..|x|] = \overline{y[1..|x| - 1]})$ AND $(z = x\overline{y[|x| - i + 1..|y|]})$;
2. $\exists i$ such that $(|x| - i + 1 > |y|)$ AND
   $(x[i..i + |y|-] = \overline{y})$ AND $(z = x)$;
3. $\exists i$ such that $(|y| - i + 1 \leq |x|)$ AND
   $(x[1..|y| - i] = \overline{y[i..|y|]})$ AND $(z = \overline{y[1..i - 1]}x)$;
4. $\exists i$ such that $(|y| - i + 1 > |x|)$ AND
   $(x = \overline{y[i..i + |x| - 1]})$ AND $(z = \overline{y[1..i - 1]}x\overline{(y[i + |x|..|y|])}\}$.

Intuitively, this corresponds to overlapping the two stranded strings so that they have a common part and completing the upper string with the complement of the lower string in positions not covered by the upper string.

Proposition 7. *There exist Watson-Crick morphisms which are not realizable through operations complying to Definition 3.*

Proof. In order to simulate Watson-Crick superposition through overlapping, we would need to assume Σ to be self-involutive, each element to be not comparable with any other and the operation & to be defined as follows:

1) $x\&\tau = x$
2) $\tau\&x = \overline{x}$
3) $x\&\overline{x} = x$
4) $x\&y = \blacktriangle$

for $x \in \Sigma$, $y \neq \overline{x}$.

However, such a definition of & would violate the non decreasing property of Proposition 2, as it would require $x \leq \overline{x}$ and $\overline{x} \leq x$. But since $\leq$ is a partial order, we would have $x = \overline{x}$, which is in general not true. □

Proposition 8. *The overlapping operation can simulate rewriting of regular grammars.*

Proof. Let $G = (NT, T, S, P)$ be a regular grammar. Let $ind : P \to N$ be an indexing of P. For each symbol $X \in NT$, we replace it with symbols (X, p) for all p such that p is the index of a rule (X, γ). For each symbol $a \in T$, we replace it with the symbol (a, p) for all p such that p is the index of a rule $(X, a\beta)$. We define an ordering on $NT \cup T$ through $(A, q) < (a, r)$ for all $(A, q) \in NT$, $(a, r) \in T$ (in such cases we also write $NT < T$). For each rule $(X \to \gamma) \in P$ of index p, we introduce in Π_0 the pointed strings $(\gamma', 1, |\ \gamma\ |)$, where γ' is obtained by replacing the symbols in γ with their indexed versions. Finally, we add to Π_0 all the elements from T and all the elements (S, q) from NT. The operation & is now defined by ($\tau\&x = x$ for all $x \in NT \cup T$, $(X, n)\&(a, n) = a$, $(X, n)\&(b, m) = \blacktriangle$ for $m \neq n$, $y\&z = \blacktriangle$ for all other cases. One observes that $L(G) = cut(proj(L(\Pi_0) \cap \Phi_{\sim\blacktriangle}))$, where $proj$ removes the indexes from the symbols in $NT \cup T$. □

As an example, consider the grammar $G = (\{S, A\}, \{a, b, c\}, S, P)$, where $P = \{(S \to aS), (S \to bA), (A \to bA), (A \to c)\}$). Applying the construction above, we obtain the set $\Pi_0 = \{(S, 1), (S, 2), (a, 1)S(1), (a, 1)(S, 2), (b, 2)(A, 3), (b, 2)(A, 4), (b, 3)(A, 3), (b, 3)(A, 4), (c, 4)\}$, where the entry and exit positions follow the convention above.

The possibility of going beyond regular grammars is constrained by the fact that overlapping cannot create intermediate positions between two adjacent ones, which is what grammars beyond regulars need to do. On the other hand, it is possible to augment alphabet elements with natural numbers, so that the space needed by a word would be precomputed. This calls for mechanisms such as those of 2-level grammars [23], where a high-level scheme is used to generate specialized versions of grammar rules, or attribute grammars, by considering in the language only symbols with certain values of attributes.

In particular, consider the set of rule schemes

$\{(S, n) \to \tau^{n-1}(A, n),$
$(A, M) \to (A, M - 1)a,$
$(A, 0) \to (SB, n),$
$(SB, n) \to (B, n - 1)\tau^{n-1},$

$(B, M) \to b(B, M-1)$,
$(B, 0) \to b, \}$
where n is instantiated to a constant and M to all possible values between 0 and n. We codify these rules into a set Π_0 as before. In particular, any string $\tau^n A$ presents the entry point on the first τ and the exit point on A, all strings Aa present entry point on a and exit point on A, the string SB presents entry and exit points in 0, the string $B\tau^{n-1}$ entry point on the last τ and exit point on B, the string bB entry point on b and exit point on B, and the string b entry and exit point on b. We define an ordering given by $NT < T$, and & as follows:

1) $\tau \& x = x$ for $x \in NT \cup T$
2) $A \& a = a$
3) $B \& b = b$
4) $x \& y = \blacktriangle$ otherwise

Now, one has $cut(proj(L(\Pi_0) \cap \phi_{\sim\blacktriangle})) = \{a^n b^n \mid n \geq 0\}$.

6. Conclusions

Computation on multidimensional words is becoming standard practice for current technology, both for multimedia applications and for representing evolution of distributed states. In general, ad hoc methods have to be devised for working on different numbers of dimensions or for modeling the semantics of such computations.

We have proposed a general setting for describing computations on multidimensional words, which can be parameterized to any number of dimensions, and which imposes some form of consistency on the contributions provided by independent agents. Agents can cooperate in the construction of words defining the result of a computation only on designated positions. We have shown the approach at work on some concrete examples, and discussed its relations with some devices from formal language theory.

Note that, while less powerful than the original proposal of gluing of pointed pictures with orientation, the operation of overlapping is still able to generate the same set of pictures as gluing, provided that we preprocess the pictures in a set Π_0 to produce a set Π_0' which contains all the rotated versions of the pictures in Π_0 plus a finite set of tiles which allow moving exit points around.

Drawing inspiration from forms of composition of different contributions in such a way that each intermediate state of the computation is "more defined" than the previous one, we obtain an interesting algebraic structure. As a matter of fact, the meet-semilattice monoid structure is a first step to build an enriched categorical setting [13, 22], which can also accommodate a logical system, very close to that associated with a topos. Such a setting has been used to model concurrent systems [12] according to the intuition by Milner [18]. The logical language corresponding to this structure can be used to filter out moves which cannot contribute to reaching a desired final state. Future work will investigate the properties of the logic defined from the enriched structure and will proceed on a more systematic basis to the exploration of the computational power of the approach, also considering its components in isolation, viz. ordering of alphabets, use of overlapping, designated positions, filters.

Acknowledgements. Work partially supported by the Italian Ministry of Research (PRIN 2006). We are happy to dedicate this work to Victor Mitrana on the occasion of his 50th birthday. We have enjoyed working with him to exploit the power of superposition and hope that this new operation will meet his interest.

References

[1] BERSTEL J., BOASSON L., *Partial words and a theorem of Fine and Wilf*, TCS, **218**, 1999, pp. 135–141.

[2] BOTTONI P., LABELLA A., *Pointed pictures*, JVLC, **18**, 2007, pp. 523–536.

[3] BOTTONI P., LABELLA A., MANCA V., MITRANA V., *Superposition based on Watson-Crick-like complementarity*, Theory of Computing Systems, **39**, 2006, pp. 503–524.

[4] BOTTONI P., MAURI G., MUSSIO P., *From strings to pictures and back*, ROMJIST, **6**, 2003, pp. 87–104.

[5] BOTTONI P., MAURI G., MUSSIO P., PĂUN G., *Grammars working on layered strings*, Acta Cybernetica, **13**, 1998, pp. 339–358.

[6] BOTTONI P., MAURI G., MUSSIO P., PĂUN G., *On the power of pictorial languages*, IJPRAI, **14**, 2000, pp. 839–858.

[7] DACEY M. F., *The syntax of a triangle and some other figures*, Patt. Rec., **2**, 1970, pp. 11–31.

[8] GIAMMARRESI D., RESTIVO A., *Two-dimensional languages*, in G. Rozenberg, A. Salomaa (eds.), Handbook of Formal Languages. Volume **III**, Springer, 1997, pp. 215–267.

[9] GRAMMATIKOPOULOU A., *Prefix picture sets and picture codes*, (`web.auth.gr/cai05/papers/21.pdf`).

[10] KALYANI T., DARE V., THOMAS D., *Local and recognizable iso picture languages*, Proc. ICONIP 2004, pp. 738–743.

[11] KARI L., PĂUN G., ROZENBERG G., SALOMAA A., YU S., *DNA computing, sticker systems, and universality*, Acta Informatica **35**, 1998, pp. 401–420.

[12] KASANGIAN S., LABELLA A., *Observational trees as models for concurrency*, MSCS, **9**, 1999, pp. 687–718.

[13] KELLY G., *Basic Concepts of Enriched Category Theory*, Number 64 in London Mathematical Society Lecture Note Series, Cambridge University Press, 1982.

[14] KIRSCH R., *Computer interpretation of English text and picture patterns*, IEEE Transactions on Electr. Computer **13**, 1964, pp. 363–376.

[15] LATTEUX M., ROBILLIARD D., SIMPLOT D., *Figures composées de pixels et monoïde inversif*, Bull. Belg. Math. Soc. **4**, 1997, pp. 89–111.

[16] LACK S., SOBOCINSKI P., *Adhesive Categories*, Proc. FOSSACS 2004, pp. 273–288.

[17] MERCER A., ROSENFELD A.,*An array grammar programming system*, Comm. ACM, **16**, 1973, pp. 299–305.

[18] MILNER R., *Communication and concurrency*, Prentice Hall International, 1989.

[19] SHAW A., *The formal picture description scheme as a basis for picture processing systems*, Information and Control **14**, 1969, pp. 9–52.

[20] SIROMONEY R., KRITHIVASAN K., *Parallel context-free grammars*, Information and Control, **24**, 1974, pp. 155–162.

[21] SIROMONEY G., SIROMONEY R., KRITHIVASAN K., *Abstract families of matrices and picture languages*, CGIP, **1**, 1972, pp. 284–307.

[22] WALTERS R., *Sheaves and Cauchy-complete categories*, Cahiers de Topologie et Geometrie Diff. **22**, 1981, pp. 283–286.

[23] VAN WIJNGAARDEN A., *The generative power of two-level grammars*, *Proc. ICALP 1974*, pp. 9–16.