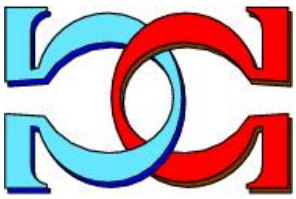
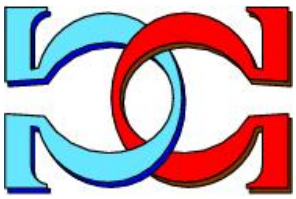
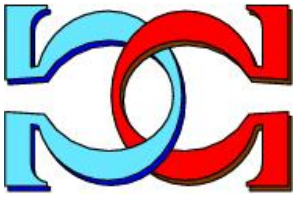


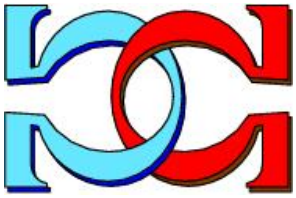
CDMTCS
Research
Report
Series

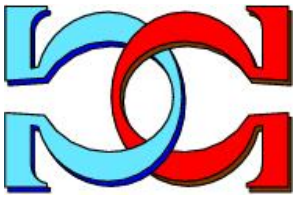


P Systems in Stereo Matching

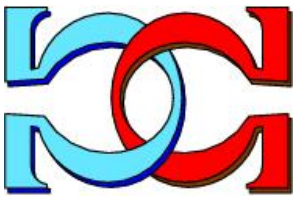


Georgy Gimel'farb
Radu Nicolescu
Sharvin Ragavan
Department of Computer Science,
University of Auckland,
Auckland, New Zealand





CDMTCS-401
April 2011



Centre for Discrete Mathematics and
Theoretical Computer Science

P Systems in Stereo Matching

Georgy Gimel'farb, Radu Nicolescu, Sharvin Ragavan

Abstract

Designing parallel versions of sequential algorithms has attracted renewed attention, due to recent hardware advances, including various general-purpose multi-core, multiple core and many-core processors, as well as special-purpose FPGA implementations. P systems consist of networks of autonomous cells, such that each cell transforms its input signals in accord with symbol-rewriting rules and feeds the output results into its immediate neighbours. Inherent intra- and inter-cell parallelism make the P systems a prospective theoretical testbed for designing parallel algorithms. This paper discusses capabilities of P systems to implement the symmetric dynamic programming algorithm for stereo matching, with due account to binocular or monocular visibility of 3D surface points.

Keywords: Parallel systems, membrane computing, stereo matching, symmetric dynamic programming stereo (SDPS).

1 Introduction

Essentially, a P system is a network of data processing cells, abstracting various features of the nervous system [3, 6, 7, 8, 9, 10]. Each cell transforms input and local symbols in accord with rewriting rules and sends some of the resulting symbols out, to its close neighbours. Rules of the same cell can be applied in parallel (if possible) and all cells work in parallel, in the synchronous mode. The underlying network is a *digraph* or a more specialized version, such as a directed acyclic graph (*dag*) or a *tree* (which is the most studied case). Advanced scenarios consider cases when cells or arcs can dynamically appear, disappear or move.

P systems provide a theoretically efficient testbed for the design of parallel versions of sequential data processing algorithms, including various image analysis tasks, such as *segmentation* [2, 1]. This paper discusses a P system based implementation of the *symmetric dynamic programming stereo* (SDPS) [5]. To the best of our knowledge, this is the first paper that introduces a P system model for stereo matching and it reveals and separates the inherently parallel and sequential processing stages of the SDPS algorithm.

SDPS searches for the maximal scanline-to-scanline similarity, in explicit or implicit Cyclopean space of (x, y)-coordinates and d-disparities [5]. The most obvious and "trivial" avenue for parallel implementation of the SDPS is common for all so-called 1D stereo matching algorithms: similarity between each pair of corresponding scanlines in a rectified (co-aligned) stereo pair of images can be maximised independently of other pairs. SDPS

offers an additional parallelisation avenue, exploited in this paper, by computing similarity scores in parallel, for all the disparities associated with each current x-coordinate. Together, both properties suggest a massively parallel 3D membrane computing framework combining the 2D parallel and 1D sequential forward-backward processing. The main part of this framework consists of a 3D (x, y, d) uniform array of similar cells, connected each to the near neighbours, along the x-coordinate and d-disparity axes. The cells are linked to image memory for simultaneous initialisation by embedding image data into each cell. Fixed data transmission links between the neighbouring cells facilitate the forward propagation of 2D processing, in parallel along the 3D array, and the backward trace, required to reconstruct the goal 3D surface, consisting of independently found 2D cross-sections, or profiles.

2 General P-Model

The basic definition of *simple P modules* in [3] covers most common P systems, such as *cell-like* (based on trees), *hyperdag* (based on dags), and *neural* P systems (based on directed graphs). This definition is further generalised, by introducing new features, which appear useful for modelling the SDPS.

Definition 1. A simple P module with duplex channels *is a system* $\Pi = (O, K, \delta)$*, where* O *is a finite non-empty alphabet of objects;* K *is a finite set of cells and* δ *is an irreflexive binary relation on* K*, representing a set of structural arcs between cells (essentially a digraph), with duplex communication capabilities.*

Each cell, $\sigma_i \in K$*, has the initial configuration* $\sigma_i = (Q_i, s_{i0}, w_{i0}, R_i)$*, and the current configuration* $\sigma_i = (Q_i, s_i, w_i, R_i)$*, where:*

- Q_i *is a finite set of states;*
- $s_{i0} \in Q_i$ *is the initial state;*
- $s_i \in Q_i$ *is the current state;*
- $w_{i0} \in O^*$ *is the initial content;*
- $w_i \in O^*$ *is the current content;*
- R_i *is a finite ordered set of multiset rewriting rules of the form:* $s\ x \rightarrow_\alpha s'\ x'\ (u)_{\beta_\gamma}$*, where* $s, s' \in Q$*,* $x, x' \in O^*$*,* $u \in O^*$*,* $\alpha \in \{\texttt{min}, \texttt{max}\}$*,* $\beta \in \{\uparrow, \downarrow, \updownarrow\}$*, and* $\gamma \in \{\texttt{one}, \texttt{spread}, \texttt{repl}\} \cup K$*. If* $u = \lambda$*, i.e. the empty multiset of objects, this rule can be abbreviated as* $s\ x \rightarrow_\alpha s'\ x'$*.*

A cell evolves by applying one or more rules, which can change its content and state and can send objects to its neighbours. For cell $\sigma_i = (Q_i, s_i, w_i, R_i)$, a rule $s\ x \rightarrow_\alpha s'\ x'\ (u)_{\beta_\gamma} \in R_i$ is applicable if $s = s_i$ and $x \subseteq w_i$. The application of a rule takes two sub-steps, after which the cell's current state s is replaced by target state s', the current content x is replaced by x', and multiset u is sent as specified by the transfer operator β_γ (as further described below). The rules are applied in the weak priority order [9],

i.e. the higher priority applicable rules are applied before the lower priority ones, and a lower priority applicable rule is applied only if it indicates the same target state as the previously applied rules.

In this paper, we use the rewriting operators $\alpha = \texttt{min}, \texttt{max}$, and the transfer operators $\beta_\gamma = \downarrow_{\texttt{repl}}, \downarrow_j, \uparrow_j$, where $\sigma_j \in K$. The rewriting operators $\alpha = \texttt{min}$ and $\alpha = \texttt{max}$ indicate that an applicable rewriting rule of R_i is applied once or as many times as possible, respectively. Multisets u represent messages, sent to digraph neighbours, up and/or down structural arcs, according to the β_γ operators. If the right-hand side of the rule contains $(u)_{\downarrow_{\texttt{repl}}}$, then, for each application of this rule, a copy of multiset u is sent to each cell in $\delta(i)$ (i.e. to each structural *child* of σ_i). If the right-hand side of the rule contains $(u)_{\downarrow_j}$, then, for each application of this rule, a copy of multiset u is sent to cell $\sigma_j \in K$, *if* $j \in \delta(i)$ (i.e. if σ_j is a structural *child* of σ_i). However, if $j \notin \delta(i)$, then message u is silently lost. Similarly, if the right-hand side of the rule contains $(u)_{\uparrow_j}$, then, for each application of this rule, a copy of multiset u is sent to cell $\sigma_j \in K$, *if* $j \in \delta^{-1}(i)$ (i.e. if σ_j is a structural *parent* of σ_i); otherwise, if $j \notin \delta^{-1}(i)$, then message u is silently lost. Other (not used in this paper) operators are described in [4].

The above definition allows each cell to have its own state and rule sets. However, we prefer scenarios when all the cells share the same state and rule set; differing only by their initial content and their neighbourhood relations. Intuitively, such cells are created identical, by a virtual "cell factory" and initialised to the same quiescent state, typically designated as s_0. A state is called *quiescent* if no rules can be applied for empty cells in this state (i.e. an empty quiescent cell does not evolve until some object appeared in this cell, e.g. was sent from this cells' neighbours). Then, the cells are allocated different positions in the structural digraph and possibly initialised with different contents.

2.0.1 Extensions.

We extend the basic simple P-module concept with the following three features, which are needed in complex scenarios, such as the SDPS algorithm.

1. Arcs can have *labels* to be used in transfer operators, instead of cell labels. For example, if k is the label of an outgoing arc $(\sigma_i, \sigma_j) \in \delta$, the occurrence of $(u)_{\downarrow_k}$ in the right-hand side of a rule indicates that message u is to be sent from cell σ_i to cell σ_j. Although not globally unique, arc labels are locally unique, i.e. unique in each local neighbourhood.

2. A rule can send several messages (not just one), each one to a different neighbour, e.g. the occurrence of $(u)_{\downarrow_k}(u')_{\downarrow_{k'}}$ in the right-hand side of a rule indicates that messages u, u' are to be sent via arcs labelled k, k', respectively.

3. A pair of neighbouring nodes can be connected by several labelled arcs (not just one), i.e. the supporting structural digraph becomes a *multigraph*, with labelled arcs.

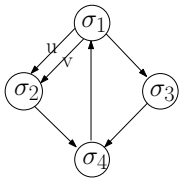


Figure 1: Example of a simple P-module, based on a multigraph with labelled arcs.

These extensions typically support scaling up the problem size, without increasing the alphabet size or the number of the rules, as several cells can reuse the same labels for their outgoing arcs.

Example 2. *Consider a simple P module* Π*, with four cells,* σ_1*,* σ_2*,* σ_3*,* σ_4*, sharing the same three states,* s_0 *(initial state),* s_1*,* s_2*, and the same ordered set of four rules:*

1. $s_0\ a \rightarrow_{\texttt{min}} s_1\ a'\ (e)_{\downarrow_{\texttt{repl}}}\ (f)_{\downarrow_u}\ (g)_{\downarrow_v}\ (h)_{\downarrow_w}$
2. $s_0\ b \rightarrow_{\texttt{max}} s_1\ b\ b'$
3. $s_0\ c \rightarrow_{\texttt{min}} s_2\ c'$
4. $s_0\ d \rightarrow_{\texttt{max}} s_1\ d'$

Assume that these four cells are arranged in the multigraph structure shown in Figure 1, where cells σ_1 and σ_2 are connected by two labelled arcs, and all cells are initially empty, except cell σ_1, which contains the object multiset ab^2cd^2. In this scenario, all rules are applicable for cell σ_1. First, rule 1 is applied once and sets the target state to s_1. Next, rule 2 is applied twice, not more, because the right-hand side objects b are produced after all rules are applied. However, rule 3 is not applied, because its right-hand side indicates a different target state, s_2. Finally, rule 4 is applied twice. In the final configuration of the system, after one step: σ_1 is in state s_1 and contains multiset $a'b^2b'^2cd'^2$; σ_2 is in state s_0 and contains multiset efg; σ_3 is in state s_0 and contains multiset e; σ_4 is in state s_0 and is empty.

3 SDPS: P system design

3.0.2 Notations.

Given an epipolar stereo pair of rectified images, let $\mathbf{C} = \mathbb{X}\mathbb{Y}.\mathbb{D}$ be a discrete space of 3D points $\mathbf{p} = (x, y, d)$ of optical (visible) surfaces reduced to the left image plane $\mathbb{X}\mathbb{Y}$ with integer planar (x, y)-coordinates $x \in \mathbb{X} = \{0, 1, \ldots, n-1\}$; $y \in \mathbb{Y} = \{0, 1, \ldots, m-1\}$ and specified by the integer disparities, $d \in \mathbb{D} = \{d_{\min}, d_{\min} + 1, \ldots, d_{\max}\}$, of corresponding points (x, y) and $(x - d, y)$ depicting $\mathbf{p}$ in the left and right image, respectively. Each 2D profile relating to a conjugate pair of scanlines with the same y-coordinate is given by a sequence of points $\mathbf{d}_y = \{(x, y, d) : x \in \mathbb{X};\ d \in \mathbb{D}\}$ such that for each two successive points (x', d') and (x, d) either $x = x' + 1$ and $d \in \{d', d' + 1\}$ or $x = x'$ and $d = d' - 1$.

Let $\mathbf{g}_{1:y} = \{g_1(x,y) : x = 0,1,\ldots,n-1\}$ and $\mathbf{g}_{2:y} = \{g_2(x,y) : x = 0,1,\ldots,n^\circ-1\}$; $y = 0,1,\ldots,m-1$, denote grey values along the conjugate scanlines (generally, $n \neq n^\circ$). Let $s \in \{B, M_1, M_2\}$ indicate visibility of a 3D point, i.e. the binocular, B, or only monocular observation by the left (M_1) or right (M_2) sensor. Under an assumed single continuous surface and the visibility constraints, only the following points (x', y, d', s') can precede the point (x, y, d, s) along each continuous profile: if $s = B$ or M_1 then $x' = x-1$ and $d' = d$ for $s' = B$ or M_2 and $x' = x-1$ and $d' = d-1$ if $s' = M_1$, and if $s = M_2$ then $x' = x$ and $d' = d+1$ for $s' = B$ or M_2.

A simplified version of the SDPS, which does not adapt for possible local contrast and offset deviations of the corresponding signals, relates the point-wise signal similarity to the absolute difference $\delta(x,y,d) = |g_1(x,y) - g_2(x-d,y)|$ between the corresponding signals for the binocularly visible point (x,y,d,B) or a regularising score $w_{\text{occl}} > 0$ for the monocularly visible ones (see [5] for more detail).

Due to unequal numbers of points in profile variants to be compared by their total signal similarity, the point-wise similarities $\varphi_y(x,d,s|x',d',s')$ are integrated between the successive points with due account of their actual shifts in Cyclopean space:
$\varphi_y(x,d,B|x',d',s') = \delta(x,y,d)$ if $s' \in \{B, M_2\}$ or $0.5\delta(x,y,d)$ if $s' = M_1$;
$\varphi_y(x,d,M_1|x',d',s') = w_{\text{occl}}$ if $s' \in \{B, M_2\}$ or $0.5w_{\text{occl}}$ if $s' = M1$ and
$\varphi_y(x,d,M_2|x',d',s') = 0.5w_{\text{occl}}$ for $s' \in \{B, M_2\}$.

3.0.3 SDPS computations.

The SDPS maximises the similarity score between the conjugate scanlines:

$$\mathbf{d}_y^* = \arg\max_{\mathbf{d}_y} \Phi_y(\mathbf{d}_y) = \sum_{i=1,2,\ldots} \varphi_y(x_i, d_i, s_i | x_{i-1}, d_{i-1}, s_{i-1}) \tag{1}$$

The *forward pass* computes potentially optimal similarity scores $F_y(x,d,s)$ and backward transitions $T_y(x,d,s)$ for each $y \in \mathbb{Y}$ by sequential pass along $x \in \mathbb{X}$ and from $d_{\max}$ to $d_{\min}$ for each $d \in \mathbb{D}$ (it can be partly parallel in $\mathbb{XD}$ plane):

$$\begin{array}{rcl} F_y(x,d,s) & = & \max\limits_{(x',d',s') \in \Omega(x,d,s)} \{F_y(x',d'.s') + \varphi_y(x,d,s|x',d',s')\} \\ T_y(x,d,s) & = & \arg \max\limits_{(x',d',s') \in \Omega(x,d,s)} \{F_y(x',d',s') + \varphi_y(x,d,s|x',d',s')\} \end{array} \tag{2}$$

where the sets of back-transitions $\Omega(x,d,s)$ follow from the visibility conditions.

The *backward pass* computes the optimal profiles $\mathbf{d}_y$ in parallel across $y \in \mathbb{Y}$ and sequentially along $x \in \mathbb{X}$:

$$\begin{array}{l} (x^* = n-1, d^*, s^*) = \arg\max_{(d,s) \in \mathbb{D} \times \mathbb{S}} \{F_y(n-1,d,s)\} \\ (x'^*, d^*, s^*) = T_y(x^*, d^*, s^*) \text{ while } x'^* > 0 \end{array} \tag{3}$$

where $n-1$ is the x-coordinate of the rightmost point of each profile variant.

3.0.4 Design issues.

Without the loss of generality, the P system has to be designed for a pair of conjugate scanlines of size n, which arguably is the most challenging parallelisation task. The solution can be further extended to a stereo pair of images in a straightforward manner, using "trivial" parallel processing of each pair of conjugate scanlines). We also take $d_{\min} = 0$, so the disparities range over the integer interval $[0, d_{\max}]$, and we encode integer numbers by repeating symbols, starting with one occurrence for zero (as in traditional λ-calculus), e.g., the base symbol a gives the following encodings: $0 \to a$, $1 \to a^2$, $2 \to a^3$, etc). We discuss our solution using the following example:

Example 3. *Consider a scenario where $n = 6$, the occlusion weight $w_{\text{occl}} = 18$, and the left and right scanlines with the following sequences of pixel values: $g_1 = \{15, 10, 30, 50, 15, 10\}$; $g_2 = \{10, 30, 50, 50, 15, 10\}$. The above SDPS algorithm finds the optimal similarity score = 36 and the profile: $\mathbf{d} = \{(0,0), (1,1), (2,1), (3,1), (3,0), (4,0), (5,0)\}$, or $d = \{0,1,1,1,0,0\}$, for $x = \{0,1,2,3,4,5\}$.*

3.0.5 Cells.

We construct a P system, Π, consisting of the following subcomponents.

- L: a list of cells, $\sigma_i^l, i \in [0, n-1]$, which represent the left scan line, initialized with the left pixel values, encoded in base x. In our example, these cells are initialized, in order, with the following multisets: x^{16}, x^{11}, x^{31}, x^{51}, x^{16}, x^{11}.

- R: a list of cells, $\sigma_i^r, i \in [0, n-1]$, which represent the right scan line, initialized with the right pixel values, encoded in base y. In our example scenario, these cells are initialized, in order, with the following multisets: y^{11}, y^{31}, y^{51}, y^{51}, y^{16}, y^{11}.

- D: a list of cells, $\sigma_i^d, i \in [0, n-1]$, which represent the disparity line, initially empty. At the end of our P program, these cells will contain the optimal disparity values, in base d. In our example, these cells will contain, in order, the following multisets: d^1, d^2, d^2, d^2, d^1, d^1.

- W: an array of cells, $\sigma_{ij}^w, i \in [0, n-1], j \in [0, d_{\max}]$, which represent the main workspace. Each cell σ_{ij} encloses three virtual subcells, respectively holding monocular-left values (M_1), binocular values (B), and monocular-right values (M_2). The initialisation of these cells is described later on, in this section.

- M: a list of cells, $\sigma_j^m, j \in [0, d_{\max}]$, which represent a secondary workspace, which identifies the optimum (minimal) score, among $d_{\max} + 1$ possible scores.

Figure 2 (left) shows all these cells, for our example.

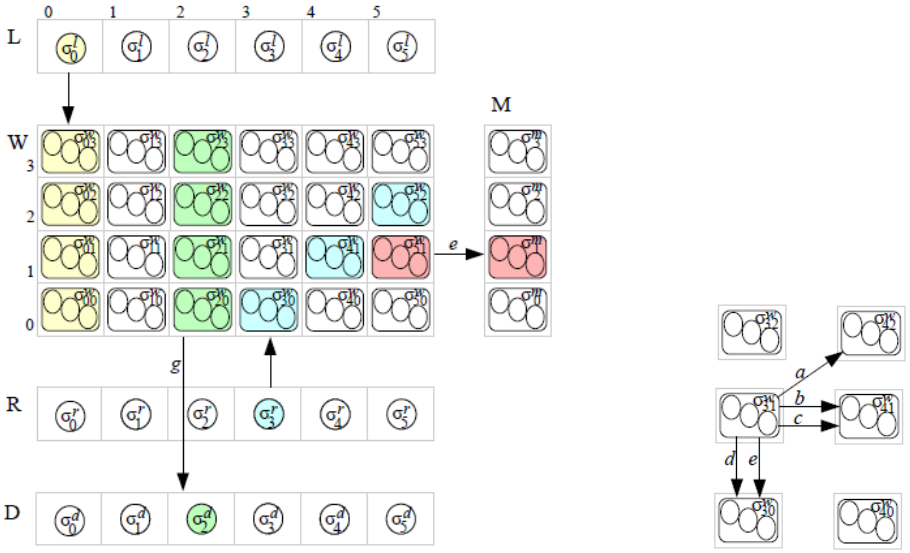


Figure 2: Cells of Π (left) and typical arcs between W-cells (right).

3.0.6 Arcs.

The cells are linked by arcs in a directed acyclic graph (dag) δ, incrementally constructed by the following enumeration.

- Each L-cell is parent of its corresponding W-column: $(\sigma^l_i, \sigma^w_{ij}) \in \delta$, $i \in [0, n-1]$, $j \in [0, d_{\max}]$.

- Each R-cell is parent of a corresponding W-diagonal (running S/W to N/E): $(\sigma^r_i, \sigma^w_{ij_i}) \in \delta$, $i \in [0, n-1]$, $j_i \in [i, \max(n, i + d_{\max})]$.

- Each W-cell before the last column and below the top row is parent of the W-cell in the N/E direction: $(\sigma^w_{ij}, \sigma^w_{i+1j+1}) \in \delta$, $i \in [0, n-2]$, $j \in [0, d_{\max} - 1]$. These arcs are *labelled* with the same dedicated symbol, a.

- $(\sigma^w_{ij}, \sigma^w_{i+1j}) \in \delta$, $i \in [0, n-2]$, $j \in [0, d_{\max}]$. These arc pairs are *labelled* with the same dedicated symbols, b and c.

- Each W-cell above the bottom row is *twice* parent of the W-cell below it (in the S direction): $(\sigma^w_{ij}, \sigma^w_{ij-1}) \in \delta$, $i \in [0, n-1]$, $j \in [1, d_{\max}]$. These arc pairs are *labelled* with the same dedicated symbols, d and e.

- Each rightmost column W-cell is parent of its corresponding M-cell: $(\sigma^w_{nj}, \sigma^m_j) \in \delta$, $j \in [0, d_{\max}]$. These arcs are all *labelled* with the same dedicated symbol, f.

- All cells in a W-column are parents of their corresponding D-cell: $(\sigma^w_{ij}, \sigma^d_i) \in \delta$, $i \in [0, n-1]$, $j \in [0, d_{\max}]$. These arcs are all *labelled* with the same dedicated symbol, g.

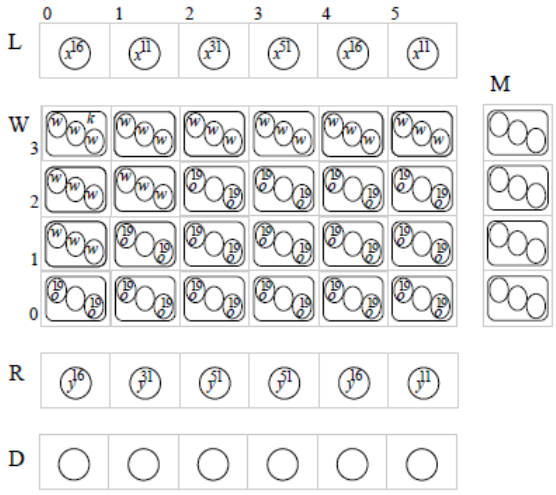


Figure 3: Initialised P system Π.

Even for our simple example, a full picture of all these arcs is impossible, because of its sheer complexity. However, we can suggest a representative fragment of these arcs. By highlighting cells that are dag neighbours, Figure 2 (left) suggests the following arcs: L-cell σ_0^l is the parent of W-cells σ_{03}^w, σ_{02}^w, σ_{01}^w, σ_{00}^w; R-cell σ_3^r is the parent of W-cells σ_{30}^w, σ_{41}^w, σ_{52}^w; W-cell σ_{51}^w is the parent of M-cells σ_1^m (arcs are labelled f); W-cells σ_{23}^w, σ_{22}^w, σ_{21}^w, σ_{20}^w are parents of D-cell σ_2^d (arcs are labelled g). Also, Figure 2 (right) shows the following arcs between W-cells: σ_{31} is parent of σ_{42}, via arc labelled a; σ_{31} is twice parent of σ_{41}, via arcs labelled b and c; σ_{31} is twice parent of σ_{30}, via arcs labelled d and e.

3.0.7 Workspace initialisation.

Some of the W-cells are initialised as follows.

- Each cell above the first S/W to N/E diagonal, σ_{ij}^w, $i \in [0, n-1]$, $j \in [i+1, d_{\max}]$, contains one copy of symbols w_l, w_b and w_r, interpreted as the infinite values of its monocular-left (M_1), binocular (B) and monocular-right (M_2) subcells, respectively.

- Each top-row cell, $\sigma_{id_{\max}}^w$, $i \in [0, n-1]$, contains one copy of symbols w_l and w_r (interpreted the same way as above).

- The top-left cell, $\sigma_{0d_{\max}}^w$, contains one copy of symbol k, which triggers the whole computation.

Figure 3 shows the initialised P system Π, just before it starts.

3.0.8 Evolution—bird's eye view.

At a very high level, the system Π evolves in two major phases, closely related to the above description of the SDPS.

Figure 4: Final scores of the subcells of the W-cells at the end of the *forward pass* phase.

First, the *forward pass* phase proceeds on slope 2 diagonals, starting from the top-left cell (initially marked with k); all cells on the same diagonal work in parallel. In our sample scenario, the computational wave moves, in order, over the diagonals $\{\sigma^w_{03}\}$, $\{\sigma^w_{02}\}$, $\{\sigma^w_{01}, \sigma^w_{13}\}$, $\{\sigma^w_{00}, \sigma^w_{12}\}$, $\{\sigma^w_{11}, \sigma^w_{23}\}$, $\{\sigma^w_{10}, \sigma^w_{22}\}$, $\{\sigma^w_{21}, \sigma^w_{33}\}$, ..., $\{\sigma^w_{40}, \sigma^w_{52}\}$, $\{\sigma^w_{51}\}$ and $\{\sigma^w_{50}\}$.

During the *forward pass*, in the general case, each cell σ^w_{ij} determines:

1. in its binocular subcell, a preliminary score, which is the absolute value of the differences between the pixel values received from σ^l_i (left scanline) and σ^r_i (right scanline);

2. in its binocular and monocular-left subcells, the sum between its previous score (cf. step 1) and the minimum value of those received via: (i) arc a: σ^w_{i-1j-1}'s monocular-left score; (ii) arc b: σ^w_{i-1j}'s binocular score; (iii) arc c: σ^w_{i-1j}'s monocular-right score;

3. in its monocular-right subcell, the sum between its previous score (cf. step 1) and the minimum score of those received via: (i) arc d: σ^w_{ij+1}'s binocular score; (ii) arc e: σ^w_{ij+1}'s monocular-right score.

Figure 4 shows the final scores of the subcells of the W-cells at the end of the *forward pass* phase.

Additionally, each W-cell keeps pointers to the origin of the minimum received scores, i.e., the label of the corresponding via arc (a, b, c, d, e), which will be used in the *backward pass* phase. These pointers are shown in Figure 5. When the wave reaches the rightmost column of W, the final scores are sent to the corresponding M-cells. These evaluate the minimum score and triggers the *backward pass*, which essentially follows back the pointers stored in the *forward pass*, during the evaluation of the minimum score. W-cells traversed during the *backward pass* send their disparity positions to the corresponding D-cells, which in the end return the problem's solution. Figure 5 shows the *backward pass* traversal using the backward pointers, and the disparity values in D-cells.

3.0.9 Rules

All cells start in the same initial quiescent state s_0 and they share the same ruleset, applied in weak priority order [9]. Later, cells begin to differentiate, by entering different

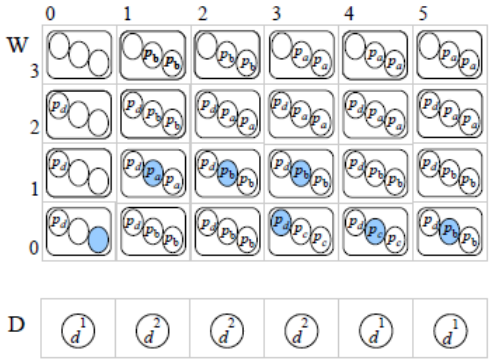


Figure 5: The *backward pass* traversal and final values of D-cells.

states (according to their contents), which implicitly partitions the initial ruleset. We outline the full ruleset by dividing it into logical fragments. Each rules fragment will be introduced by one or more expressions of type $s \overset{X:k}{\Longrightarrow} s'$, indicating a set of rules which transform the state of X cells, from s into s', in k P steps.

$\mathbf{s_0} \overset{L,R:1}{\Longrightarrow} \mathbf{s_{99}}$, $\mathbf{s_0} \overset{W,M,D:1}{\Longrightarrow} \mathbf{s_1}$. L-cells and R-cells transfer their pixel values to their corresponding W children cells and transit to state s_{99}; all other cells, i.e. W, M and D, transit to state s_1; everything in one P step.

1. $s_0\ x \rightarrow_{\texttt{max}} s_{99}\ x\ (l_b)_{\downarrow}$
2. $s_0\ y \rightarrow_{\texttt{max}} s_{99}\ y\ (r_b)_{\downarrow}$
3. $s_0 \rightarrow_{\texttt{min}} s_1$

$\mathbf{s_1} \overset{W:2}{\Longrightarrow} \mathbf{s_4}$, $\mathbf{s_1} \overset{M,D:1}{\Longrightarrow} \mathbf{s_{27}}$. Each W-cell computes the absolute value of the differences between the received left and right pixel values and transits to state s_4, in four P steps. Each other cell, i.e. M and D, transits to state s_{27}, in one P step.

1. $s_1\ w_b \rightarrow_{\texttt{min}} s_2\ w_b$
2. $s_1\ l_b\ r_b \rightarrow_{\texttt{min}} s_3\ z_b$
3. $s_1\ l_b\ r_b \rightarrow_{\texttt{max}} s_3$
4. $s_1\ l_b \rightarrow_{\texttt{max}} s_3\ z_b$
5. $s_1\ r_b \rightarrow_{\texttt{max}} s_3\ z_b$
6. $s_1\ l_b \rightarrow_{\texttt{max}} s_2$
7. $s_1\ r_b \rightarrow_{\texttt{max}} s_2$
8. $s_1 \rightarrow_{\texttt{min}} s_{27}$
9. $s_2 \rightarrow_{\texttt{min}} s_4$
10. $s_3 \rightarrow_{\texttt{min}} s_4$

$\mathbf{s_4} \overset{\sigma^w_{0d_{\max}}:2}{\Longrightarrow} \mathbf{s_7}$, $\mathbf{s_4} \overset{W\setminus\sigma^w_{0d_{\max}}:1+\cdots+1}{\Longrightarrow} \mathbf{s_8}$. The top-left W-cell, $\sigma^w_{0d_{\max}}$, which contains the initial trigger symbol k, triggers the *forward* phase and transits to s_7, in two P steps; all other W-cells linger into a intermediary quiescent state, s_6, until symbol k is received, and then transit to s_8, in two plus an unspecified number of P steps.

1. $s_4\ k \rightarrow_{\texttt{min}} s_5\ k$
2. $s_4 \rightarrow_{\texttt{min}} s_6$
3. $s_5\ o_l \rightarrow_{\texttt{max}} s_7\ u_l$
4. $s_5\ o_r \rightarrow_{\texttt{max}} s_7\ u_r$

5. $s_5\ z_b \rightarrow_{\texttt{max}} s_7\ z_b$
6. $s_5 \rightarrow_{\texttt{min}} s_7$
7. $s_6\ k \rightarrow_{\texttt{min}} s_8\ k$

$\mathbf{s_7} \overset{W_k:1}{\Longrightarrow} \mathbf{s_{26}}$. W-cells containing trigger symbol k transfer values to their neighbouring cells, via arcs a, b, c, d, e; trigger symbol k is also transferred, via arcs a and d (this is enough to ensure the proper *forward pass* workflow).

1. $s_7\ k \rightarrow_{\texttt{min}} s_{26}\ (k)_{\downarrow_a}\ (k)_{\downarrow_d}$
2. $s_7\ k \rightarrow_{\texttt{max}} s_{26}$
3. $s_7\ u_l \rightarrow_{\texttt{max}} s_{26}\ u_l\ (a)_{\downarrow_a}$
4. $s_7\ u_b \rightarrow_{\texttt{max}} s_{26}\ u_b\ (b)_{\downarrow_b}\ (d)_{\downarrow_d}$
5. $s_7\ u_r \rightarrow_{\texttt{max}} s_{26}\ u_r\ (c)_{\downarrow_c}\ (e)_{\downarrow_e}$
6. $s_7\ w_l \rightarrow_{\texttt{min}} s_{26}\ w_l\ (w^a)_{\downarrow_a}$
7. $s_7\ w_b \rightarrow_{\texttt{min}} s_{26}\ w_b\ (w^b)_{\downarrow_b}\ (w^d)_{\downarrow_d}$
8. $s_7\ w_r \rightarrow_{\texttt{min}} s_{26}\ w_r\ (w^c)_{\downarrow_c}\ (w^e)_{\downarrow_e}$

$\mathbf{s_8} \overset{W:3}{\Longrightarrow} \mathbf{s_{12}}$. W-cells compute the minimum value of those received via arcs d and e, storing the proper *backward pointer*, p_d or p_e.

1. $s_8\ w^d\ w^e \rightarrow_{\texttt{min}} s_{10}\ w^d\ E$
2. $s_8\ w^d \rightarrow_{\texttt{min}} s_9\ D$
3. $s_8\ w^e \rightarrow_{\texttt{min}} s_9\ E$
4. $s_8\ d\ e \rightarrow_{\texttt{max}} s_{10}\ i$
5. $s_8\ d \rightarrow_{\texttt{min}} s_{10}\ D$
6. $s_8\ d \rightarrow_{\texttt{max}} s_{10}$
7. $s_8\ e \rightarrow_{\texttt{min}} s_{10}\ E$
8. $s_8\ e \rightarrow_{\texttt{max}} s_{10}$
9. $s_8 \rightarrow_{\texttt{min}} s_{10}$
10. $s_9\ d \rightarrow_{\texttt{max}} s_{11}\ i$
11. $s_9\ e \rightarrow_{\texttt{max}} s_{11}\ i$
12. $s_{10} \rightarrow_{\texttt{min}} s_{11}$
13. $s_{11}\ D\ E \rightarrow_{\texttt{min}} s_{12}\ p_d$
14. $s_{11}\ D \rightarrow_{\texttt{min}} s_{12}\ p_e$
15. $s_{11}\ E \rightarrow_{\texttt{min}} s_{12}\ p_d$
16. $s_{11} \rightarrow_{\texttt{min}} s_{12}$

$\mathbf{s_{12}} \overset{W:4}{\Longrightarrow} \mathbf{s_{19}}$. W-cells compute the minimum value of those received via arcs a, b and c, storing the proper *backward pointer*, p_a, p_b or p_c.

1. $s_{12}\ w^a\ w^b\ w^c \rightarrow_{\texttt{min}} s_{16}\ w^b\ A\ C$
2. $s_{12}\ w^a \rightarrow_{\texttt{min}} s_{13}\ A$
3. $s_{12}\ w^b \rightarrow_{\texttt{min}} s_{13}\ B$
4. $s_{12}\ w^c \rightarrow_{\texttt{min}} s_{13}\ C$
5. $s_{12}\ a\ b\ c \rightarrow_{\texttt{max}} s_{16}\ j$
6. $s_{12} \rightarrow_{\texttt{min}} s_{13}$
7. $s_{12}\ a \rightarrow_{\texttt{min}} s_{16}\ A$
8. $s_{12}\ a \rightarrow_{\texttt{max}} s_{16}$
9. $s_{12}\ b \rightarrow_{\texttt{min}} s_{16}\ B$
10. $s_{12}\ b \rightarrow_{\texttt{max}} s_{16}$
11. $s_{12}\ c \rightarrow_{\texttt{min}} s_{16}\ C$
12. $s_{12}\ c \rightarrow_{\texttt{max}} s_{16}$
13. $s_{13}\ a\ b \rightarrow_{\texttt{max}} s_{14}\ j$
14. $s_{13}\ b\ c \rightarrow_{\texttt{max}} s_{14}\ j$

15. $s_{13}\ a \rightarrow_{\texttt{max}} s_{15}\ j$
16. $s_{13}\ b \rightarrow_{\texttt{max}} s_{15}\ j$
17. $s_{13}\ c \rightarrow_{\texttt{max}} s_{15}\ j$
18. $s_{13}\ a \rightarrow_{\texttt{min}} s_{14}\ A$
19. $s_{13}\ a \rightarrow_{\texttt{max}} s_{14}$
20. $s_{13}\ b \rightarrow_{\texttt{min}} s_{14}\ B$
21. $s_{13}\ b \rightarrow_{\texttt{max}} s_{14}$
22. $s_{13}\ c \rightarrow_{\texttt{min}} s_{14}\ C$
23. $s_{13}\ c \rightarrow_{\texttt{max}} s_{14}$
24. $s_{13} \rightarrow_{\texttt{min}} s_{14}$
25. $s_{14} \rightarrow_{\texttt{min}} s_{17}$
26. $s_{15} \rightarrow_{\texttt{min}} s_{17}$
27. $s_{16} \rightarrow_{\texttt{min}} s_{17}$
28. $s_{17} \rightarrow_{\texttt{min}} s_{18}$
29. $s_{18}\ A\ B\ C \rightarrow_{\texttt{min}} s_{19}\ p_a$
30. $s_{18}\ A\ B \rightarrow_{\texttt{min}} s_{19}\ p_c$
31. $s_{18}\ A\ C \rightarrow_{\texttt{min}} s_{19}\ p_b$
32. $s_{18}\ B\ C \rightarrow_{\texttt{min}} s_{19}\ p_a$
33. $s_{18}\ A \rightarrow_{\texttt{min}} s_{19}\ p_b$
34. $s_{18}\ B \rightarrow_{\texttt{min}} s_{19}\ p_a$
35. $s_{18}\ C \rightarrow_{\texttt{min}} s_{19}\ p_b$
36. $s_{18} \rightarrow_{\texttt{min}} s_{19}$

$\mathbf{s_{19}} \overset{W:5}{\Longrightarrow} \mathbf{s_7}$. Each W-cell calculates the sum between its previous score (the absolute value of pixel values differences) and the selected minimum values from the previous stages. The outgoing state at this stage is s_7, which allows the cells to apply the rules to transfer values to neighbours in subsequent steps. Table 1 shows the traces of the *forward pass* phase of the first three and last three diagonals of the W-cells.

1. $s_{19}\ w^b \rightarrow_{\texttt{min}} s_{20}\ w_l\ w_b$
2. $s_{19}\ w_l \rightarrow_{\texttt{min}} s_{20}\ w_l$
3. $s_{19}\ w_b \rightarrow_{\texttt{min}} s_{20}\ w_b$
4. $s_{19}\ o_l \rightarrow_{\texttt{min}} s_{21}$
5. $s_{19}\ z_b \rightarrow_{\texttt{min}} s_{21}$
6. $s_{19}\ j \rightarrow_{\texttt{min}} s_{21}$
7. $s_{19}\ o_l \rightarrow_{\texttt{max}} s_{21} u_l$
8. $s_{19}\ z_b \rightarrow_{\texttt{max}} s_{21} u_b$
9. $s_{19}\ j \rightarrow_{\texttt{max}} s_{21} u_l\ u_b$
10. $s_{19} \rightarrow_{\texttt{min}} s_{21} u_l$
11. $s_{19} \rightarrow_{\texttt{min}} s_{21} u_b$
12. $s_{20}\ w_l\ w_l \rightarrow_{\texttt{min}} s_{22}\ w_l$
13. $s_{20}\ w_b\ w_b \rightarrow_{\texttt{min}} s_{22}\ w_b$
14. $s_{20}\ o_l \rightarrow_{\texttt{max}} s_{22}$
15. $s_{20}\ z_b \rightarrow_{\texttt{max}} s_{22}$
16. $s_{20} \rightarrow_{\texttt{min}} s_{22}$
17. $s_{21} \rightarrow_{\texttt{min}} s_{22}$
18. $s_{22}\ w_d \rightarrow_{\texttt{min}} s_{23}\ w_r$
19. $s_{22}\ o_r \rightarrow_{\texttt{min}} s_{24}\ o_r$
20. $s_{22}\ i \rightarrow_{\texttt{min}} s_{24}\ i$
21. $s_{22} \rightarrow_{\texttt{min}} s_{23}$
22. $s_{23}\ w_r\ w_r \rightarrow_{\texttt{min}} s_{25}\ w_r$
23. $s_{23}\ o_r \rightarrow_{\texttt{max}} s_{25}$
24. $s_{23} \rightarrow_{\texttt{min}} s_{25}$

25. $s_{24}\ o_r \rightarrow_{\texttt{min}} s_{25}$
26. $s_{24}\ i \rightarrow_{\texttt{min}} s_{25}$
27. $s_{24}\ o_r \rightarrow_{\texttt{max}} s_{25}\ u_r$
28. $s_{24}\ i \rightarrow_{\texttt{max}} s_{25}\ u_r$
29. $s_{24} \rightarrow_{\texttt{min}} s_{25}\ u_r$
30. $s_{25} \rightarrow_{\texttt{min}} s_7$

$\mathbf{s_{26}} \overset{W:1}{\Longrightarrow} \mathbf{s_{98}}$. Recall that W-cells transit to s_{26} upon transferring their values to neighbouring cells (see rules for s_7). At s_{26}, each W-cell transits to a quiescent state, s_{98}, where it waits for the *backward pass* phase. If an arc labelled f is available, i.e. if the cell is in the rightmost W-column, the cell transfers its current values, monocular left, binocular and monocular right, to its corresponding M-cell.

1. $s_{26}\ u_l \rightarrow_{\texttt{max}} s_{98}\ u_l\ (u_l)_{\downarrow_f}$
2. $s_{26}\ u_b \rightarrow_{\texttt{max}} s_{98}\ u_b\ (u_b)_{\downarrow_f}$
3. $s_{26}\ u_r \rightarrow_{\texttt{max}} s_{98}\ u_r\ (u_r)_{\downarrow_f}$
4. $s_{26}\ w_l \rightarrow_{\texttt{min}} s_{98}\ w_l\ (w_l)_{\downarrow_f}$
5. $s_{26}\ w_b \rightarrow_{\texttt{min}} s_{98}\ w_b\ (w_b)_{\downarrow_f}$
6. $s_{26}\ w_r \rightarrow_{\texttt{min}} s_{98}\ w_r\ (w_r)_{\downarrow_f}$

$\mathbf{s_{27}} \overset{M:11}{\Longrightarrow} \mathbf{s_{40}}$. M-cells determine the overall minimum of all the values received via arcs f.

1. $s_{27}\ w_r \rightarrow_{\texttt{min}} s_{28}\ w_r$
2. $s_{27}\ n \rightarrow_{\texttt{min}} s_{28}\ n\ (h)_{\uparrow_h}$
3. $s_{28}\ u_l\ u_b\ u_r \rightarrow_{\texttt{max}} s_{29}\ m$
4. $s_{28}\ u_b \rightarrow_{\texttt{min}} s_{30}\ m\ L\ R\ N$
5. $s_{28}\ u_b \rightarrow_{\texttt{max}} s_{30}\ m$
6. $s_{29}\ u_l \rightarrow_{\texttt{min}} s_{31}\ L$
7. $s_{29}\ u_b \rightarrow_{\texttt{min}} s_{31}\ B$
8. $s_{29}\ u_r \rightarrow_{\texttt{min}} s_{31}\ R$
9. $s_{29}\ u_l \rightarrow_{\texttt{max}} s_{31}$
10. $s_{29}\ u_b \rightarrow_{\texttt{max}} s_{31}$
11. $s_{29}\ u_r \rightarrow_{\texttt{max}} s_{31}$
12. $s_{30}\ m \rightarrow_{\texttt{max}} s_{33}\ q$
13. $s_{31}\ m\ n \rightarrow_{\texttt{max}} s_{32}\ q$
14. $s_{32}\ m \rightarrow_{\texttt{min}} s_{33}\ M$
15. $s_{32}\ n \rightarrow_{\texttt{min}} s_{33}\ N$
16. $s_{32}\ m \rightarrow_{\texttt{max}} s_{33}$
17. $s_{32}\ n \rightarrow_{\texttt{max}} s_{33}$
18. $s_{33}\ q \rightarrow_{\texttt{max}} s_{34}\ q\ (n)_{\downarrow_h}$
19. $s_{34} \rightarrow_{\texttt{min}} s_{35}$
20. $s_{35}\ h \rightarrow_{\texttt{min}} s_{36}\ h$
21. $s_{35} \rightarrow_{\texttt{min}} s_{37}\ f$
22. $s_{36}\ f \rightarrow_{\texttt{min}} s_{37}\ f$
23. $s_{37}\ M \rightarrow_{\texttt{min}} s_{38}\ f$
24. $s_{37}\ N \rightarrow_{\texttt{min}} s_{39}\ N$
25. $s_{38}\ f \rightarrow_{\texttt{max}} s_{99}\ (f)_{\downarrow_h}$
26. $s_{39}\ L\ B \rightarrow_{\texttt{min}} s_{40}\ r$
27. $s_{39}\ L\ R \rightarrow_{\texttt{min}} s_{40}\ b$
28. $s_{39}\ B\ R \rightarrow_{\texttt{min}} s_{40}\ l$
29. $s_{39}\ L \rightarrow_{\texttt{min}} s_{40}\ b$
30. $s_{39}\ B \rightarrow_{\texttt{min}} s_{40}\ r$
31. $s_{39}\ R \rightarrow_{\texttt{min}} s_{40}\ b$

$\mathbf{s_{40}} \overset{M:1}{\Longrightarrow} \mathbf{s_{99}}$. The disparity at the minimum value is then transferred to the W-cell where the minimum value originated from. Symbols f trigger the *backward pass* phase.

1. $s_{40}\ l \rightarrow_{\texttt{min}} s_{99}\ l\ (l)_{\uparrow_f}$
2. $s_{40}\ b \rightarrow_{\texttt{min}} s_{99}\ b\ (b)_{\uparrow_f}$
3. $s_{40}\ r \rightarrow_{\texttt{min}} s_{99}\ r\ (r)_{\uparrow_f}$
4. $s_{40}\ f \rightarrow_{\texttt{max}} s_{99}\ f\ (f)_{\uparrow_f}$

$\mathbf{s_{98}} \overset{W:1}{\Longrightarrow} \mathbf{s_{41}}$. W-cells are activated by (at least) one symbol f and transit to s_{41}.

1. $s_{98}\ f \rightarrow_{\texttt{min}} s_{41}\ f$

$\mathbf{s_{41}} \overset{W:3}{\Longrightarrow} \mathbf{s_{99}}$. This is the *backward pass*, along the selected optimal backward path. Each current W-cell on this path uses its *backward pointers*, stored during the *forward pass* (see rule fragments starting from states s_8 and s_{12}), to activate the preceding optimal W-cell, in the backwards sense. At the same time, the current cell sends its disparity position to its corresponding D-cell, via arc g.

1. $s_{41}\ l \rightarrow_{\texttt{min}} s_{42}\ l$
2. $s_{41}\ b \rightarrow_{\texttt{min}} s_{42}\ b$
3. $s_{41}\ r \rightarrow_{\texttt{min}} s_{43}\ r$
4. $s_{41}\ f \rightarrow_{\texttt{max}} s_{42}\ f\ (d)_{\downarrow_g}$
5. $s_{42}\ p_a\ f \rightarrow_{\texttt{min}} s_{44}\ p_a\ (l)_{\downarrow_a}$
6. $s_{42}\ p_b \rightarrow_{\texttt{min}} s_{45}\ p_b\ (b)_{\downarrow_b}$
7. $s_{42}\ p_c \rightarrow_{\texttt{min}} s_{46}\ p_c\ (r)_{\downarrow_c}$
8. $s_{43}\ p_d \rightarrow_{\texttt{min}} s_{47}\ p_d\ f\ (b)_{\downarrow_d}$
9. $s_{43}\ p_e \rightarrow_{\texttt{min}} s_{48}\ p_e\ f\ (r)_{\downarrow_e}$
10. $s_{44}\ f \rightarrow_{\texttt{max}} s_{99}\ (f)_{\downarrow_a}$
11. $s_{45}\ f \rightarrow_{\texttt{max}} s_{99}\ (f)_{\downarrow_b}$
12. $s_{46}\ f \rightarrow_{\texttt{max}} s_{99}\ (f)_{\downarrow_c}$
13. $s_{47}\ f \rightarrow_{\texttt{max}} s_{99}\ (f)_{\downarrow_d}$
14. $s_{48}\ f \rightarrow_{\texttt{max}} s_{99}\ (f)_{\downarrow_e}$

3.0.10 Runtime complexity.

The asymptotic runtime complexity of our P model, which is arguably optimal, is summarized by Theorem 4. The proof follows from the structure and their execution of the above rules in the P system. In contrast, the best implementation on existing parallel hardware is limited by $O(nd/q)$, where q is the number of available processors.

Theorem 4. *The P system described in Section 3 completes in $O(n + d)$ P steps.*

4 Conclusion

In this paper, we presented a massively parallel P-model for implementing a critical part of the SDPS algorithm. Our solution is based on simple P-modules, with extensions which allow our model to adapt to complex cases such as SDPS. Our model processes in parallel all potentially optimal similarity scores that trace candidate decisions, for all the disparities associated with each current x-coordinate. We plan to further extend the solution, from pairs of scanlines to stereo pairs of full images. To avoid complex and lengthy arguments, in this paper we did not discuss how all the cells have been created and initialised. We also plan to investigate an advanced model, which starts as one single "ur-cell" and then grows, until it reaches the proper size and shape, required by the SDPS solution.

References

[1] Carnero, J., Díaz-Pernil, D., Molina-Abril, H.: Image segmentation inspired by cellular models using hardware programming. In: Díaz, R.G., Jurado, P.R. (eds.) 3rd International Workshop on Computational Topology in Image Context. pp. 143–150 (2010)

[2] Christinal, H.A., Díaz-Pernil, D., Jurado, P.R.: P systems and computational algebraic topology. Mathematical and Computer Modelling 52(11-12), 1982–1996 (2010)

[3] Dinneen, M.J., Kim, Y.B., Nicolescu, R.: A faster P solution for the Byzantine agreement problem. In: Gheorghe, M., Hinze, T., Păun, G. (eds.) Conference on Membrane Computing. Lecture Notes in Computer Science, vol. 6501, pp. 175–197. Springer-Verlag, Berlin Heidelberg (2010)

[4] Dinneen, M.J., Kim, Y.B., Nicolescu, R.: P systems and the Byzantine agreement. Journal of Logic and Algebraic Programming 79(6), 334–349 (2010)

[5] Gimel'farb, G.L.: Probabilistic regularisation and symmetry in binocular dynamic programming stereo. Pattern Recognition Letters 23(4), 431–442 (2002)

[6] Nicolescu, R., Dinneen, M.J., Kim, Y.B.: Towards structured modelling with hyperdag P systems. International Journal of Computers, Communications and Control 2, 209–222 (2010)

[7] Paun, G.: Computing with membranes. J. Comput. Syst. Sci. 61(1), 108–143 (2000)

[8] Păun, G.: Membrane Computing: An Introduction. Springer-Verlag New York, Inc., Secaucus, NJ, USA (2002)

[9] Păun, G.: Introduction to membrane computing. In: Ciobanu, G., Pérez-Jiménez, M.J., Păun, G. (eds.) Applications of Membrane Computing, pp. 1–42. Natural Computing Series, Springer-Verlag (2006)

Table 1: The traces of the *forward pass* phase of the first three and last three diagonals (rounds) of the W-cells.

Round	0	1	2	3	n-2	n-1	n
Step	σ_{03}	σ_{02}	σ_{13}	σ_{12}	σ_{52}	σ_{51}	σ_{50}
			σ_{01}	σ_{00}	σ_{40}		
0	$s_4\ w_l w_b w_r$	$s_6\ w_l w_b w_r$	$s_6\ w_l w_b w_r$	$s_6\ w_l w_b w_r$	$s_6\ o_l^{19} z_b^{41} o_r^{19}$	$s_6\ o_l^{19} z_b^{6} o_r^{19}$	$s_6\ o_l^{19} z_b o_r^{19}$
			$s_6\ w_l w_b w_r$	$s_6\ o_l^{19} z_b^{6} o_r^{19}$	$s_6\ o_l^{19} z_b o_r^{19}$		
1	$s_5\ w_l w_b w_r$	$s_6\ w_l w_b w_r w^d w^e$	$s_6\ w_l w_b w_r w^a w^b w^c$	$s_6\ w_l w_b w_r w^a w^b w^c w^d w^e$	$s_6\ o_l^{19} z_b^{41} o_r^{19} a^{37} b^{72} c^{88} d^{95} w^e$	$s_6\ o_l^{19} z_b^{6} o_r^{19} a^{55} b^{54} c^{90} d^{77} e^{113}$	$s_6\ o_l^{19} z_b o_r^{19} b^{37} c^{72} d^{59} e^{95}$
			$s_6\ w_l w_b w_r w^d w^e$	$s_6\ o_l^{19} z_b^{6} o_r^{19} w^d w^e$	$s_6\ o_l^{19} z_b o_r^{19} b^{37} c^{37} d^{54} e^{90}$		
2	$s_7\ w_l w_b w_r$	$s_8\ w_l w_b w_r w^d w^e$	$s_8\ w_l w_b w_r w^a w^b w^c$	$s_8\ w_l w_b w_r w^a w^b w^c w^d w^e$	$s_8\ o_l^{19} z_b^{41} o_r^{19} a^{37} b72 c^{88} d^{95} w^e$	$s_8\ o_l^{19} z_b^{6} o_r^{19} a^{55} b^{54} c^{90} d^{77} e^{113}$	$s_8\ o_l^{19} z_b o_r^{19} b^{37} c^{72} d^{59} e^{95}$
			$s_8\ w_l w_b w_r w^d w^e$	$s_8\ o_l^{19} z_b^{6} o_r^{19} w^d w_e$	$s_8\ o_l^{19} z_b o_r^{19} b^{37} c^{37} d^{54} e^{90}$		
3	$s_{26}\ w_l w_b w_r$	$s_{10}\ w_l w_b w_r w^d E$	$s_{10}\ w_l w_b w_r w^a w^b w^c$	$s_{10}\ w_l w_b w_r w^a w^b w^c w^d E$	$s_9\ o_l^{19} z_b^{41} o_r^{19} a^{37} b72 c^{88} d^{95} E$	$s_{10}\ o_l^{19} z_b^{6} o_r^{19} a^{55} b^{54} c^{90} i^{77} E$	$s_{10}\ o_l^{19} z_b o_r^{19} b^{37} c^{72} i^{59} E$
			$s_{10}\ w_l w_b w_r w^d E$	$s_{10}\ o_l^{19} z_b^{6} o_r^{19} w^d E$	$s_{10}\ o_l^{19} z_b o_r^{19} b^{37} c^{37} i^{54} E$		
4		$s_{11}\ w_l w_b w_r w^d E$	$s_{11}\ w_l w_b w_r w^a w^b w^c$	$s_{11}\ w_l w_b w_r w^a w^b w^c w^d E$	$s_{11}\ o_l^{19} z_b^{41} o_r^{19} a^{37} b72 c^{88} i^{95} E$	$s_{11}\ o_l^{19} z_b^{6} o_r^{19} a^{55} b^{54} c^{90} i^{77} E$	$s_{11}\ o_l^{19} z_b o_r^{19} b^{37} c^{72} i^{59} E$
			$s_{11}\ w_l w_b w_r w^d E$	$s_{11}\ o_l^{19} z_b^{6} o_r^{19} w^d E$	$s_{11}\ o_l^{19} z_b o_r^{19} b^{37} c^{37} i^{54} E$		
5		$s_{12}\ w_l w_b w_r w^d p_d$	$s_{12}\ w_l w_b w_r w^a w^b w^c$	$s_{12}\ w_l w_b w_r w^a w^b w^c w^d p_d$	$s_{12}\ o_l^{19} z_b^{41} o_r^{19} a^{37} b72 c^{88} i^{95} p_d$	$s_{12}\ o_l^{19} z_b^{6} o_r^{19} a^{55} b^{54} c^{90} i^{77} p_d$	$s_{12}\ o_l^{19} z_b o_r^{19} b^{37} c^{72} i^{59} p_d$
			$s_{12}\ w_l w_b w_r w^d p_d$	$s_{12}\ o_l^{19} z_b^{6} o_r^{19} w^d p_d$	$s_{12}\ o_l^{19} z_b o_r^{19} b^{37} c^{37} i^{54} p_d$		
6		$s_{13}\ w_l w_b w_r w^d p_d$	$s_{16}\ w_l w_b w_r w^b AC$	$s_{16}\ w_l w_b w_r w^b w^d p_d AC$	$s_{16}\ o_l^{19} z_b^{41} o_r^{19} i^{95} j^{37} p_d BC$	$s_{16}\ o_l^{19} z_b^{6} o_r^{19} i^{77} j^{54} p_d AC$	$s_{13}\ o_l^{19} z_b o_r^{19} b^{37} c^{72} i^{59} p_d$
			$s_{13}\ w_l w_b w_r w^d p_d$	$s_{13}\ o_l^{19} z_b^{6} o_r^{19} w^d p_d$	$s_{13}\ o_l^{19} z_b o_r^{19} b^{37} c^{37} i^{54} p_d$		
7		$s_{17}\ w_l w_b w_r w^d p_d$	$s_{17}\ w_l w_b w_r w^b AC$	$s_{17}\ w_l w_b w_r w^b w^d p_d AC$	$s_{17}\ o_l^{19} z_b^{41} o_r^{19} i^{95} j^{37} p_d BC$	$s_{17}\ o_l^{19} z_b^{6} o_r^{19} i^{77} j^{54} p_d AC$	$s_{17}\ o_l^{19} z_b o_r^{19} i^{59} j^{37} p_d C$
			$s_{17}\ w_l w_b w_r w^d p_d$	$s_{17}\ o_l^{19} z_b^{6} o_r^{19} w^d p_d$	$s_{17}\ o_l^{19} z_b o_r^{19} i^{54} j^{37} p_d B$		
8		$s_{18}\ w_l w_b w_r w^d p_d$	$s_{18}\ w_l w_b w_r w^b AC$	$s_{18}\ w_l w_b w_r w^b w^d p_d AC$	$s_{18}\ o_l^{19} z_b^{41} o_r^{19} i^{95} j^{37} p_d BC$	$s_{18}\ o_l^{19} z_b^{6} o_r^{19} i^{77} j^{54} p_d AC$	$s_{18}\ o_l^{19} z_b o_r^{19} i^{59} j^{37} p_d C$
			$s_{18}\ w_l w_b w_r w^d p_d$	$s_{18}\ o_l^{19} z_b^{6} o_r^{19} w^d p_d$	$s_{18}\ o_l^{19} z_b o_r^{19} i^{54} j^{37} p_d B$		
9		$s_{19}\ w_l w_b w_r w^d p_d$	$s_{19}\ w_l w_b w_r w^b p_b$	$s_{19}\ w_l w_b w_r w^b w^d p_b p_d$	$s_{19}\ o_l^{19} z_b^{41} o_r^{19} i^{95} j^{37} p_a p_d$	$s_{19}\ o_l^{19} z_b^{6} o_r^{19} i^{77} j^{54} p_b p_d$	$s_{19}\ o_l^{19} z_b o_r^{19} i^{59} j^{37} p_b p_d$
			$s_{19}\ w_l w_b w_r w^d p_d$	$s_{19}\ o_l^{19} z_b^{6} o_r^{19} w^d p_d$	$s_{19}\ o_l^{19} z_b o_r^{19} i^{54} j^{37} p_b p_d$		
10		$s_{20}\ w_l w_b w_r w^d p_d$	$s_{20}\ w_l^2 w_b^2 w_r p_b$	$s_{20}\ w_l^2 w_b^2 w_r w^d p_b p_d$	$s_{21}\ u_l^{55} u_b^{77} o_r^{19} i^{95} p_a p_d$	$s_{21}\ u_l^{72} u_b^{59} o_r^{19} i^{77} p_b p_d$	$s_{21}\ u_l^{55} u_b^{37} o_r^{19} i^{59} p_b p_d$
			$s_{20}\ w_l w_b w_r w^d p_d$	$s_{21}\ u_l^{19} u_b^{6} o_r^{19} w^d p_d$	$s_{21}\ u_l^{55} u_b^{37} o_r^{19} i^{54} p_b p_d$		
11		$s_{22}\ w_l w_b w_r w^d p_d$	$s_{22}\ w_l w_b w_r p_b$	$s_{22}\ w_l w_b w_r w^d p_b p_d$	$s_{22}\ u_l^{55} u_b^{77} o_r^{19} i^{95} p_a p_d$	$s_{22}\ u_l^{72} u_b^{59} o_r^{19} i^{77} p_b p_d$	$s_{22}\ u_l^{55} u_b^{37} o_r^{19} i^{59} p_b p_d$
			$s_{22}\ w_l w_b w_r w^d p_d$	$s_{22}\ u_l^{19} u_b^{6} o_r^{19} w^d p_d$	$s_{22}\ u_l^{55} u_b^{37} o_r^{19} i^{54} p_b p_d$		
12		$s_{23}\ w_l w_b w_r^2 p_d$	$s_{23}\ w_l w_b w_r p_b$	$s_{23}\ w_l w_b w_r^2 p_b p_d$	$s_{24}\ u_l^{55} u_b^{77} o_r^{19} i^{95} p_a p_d$	$s_{24}\ u_l^{72} u_b^{59} o_r^{19} i^{77} p_b p_d$	$s_{24}\ u_l^{55} u_b^{37} o_r^{19} i^{59} p_b p_d$
			$s_{23}\ w_l w_b w_r^2 p_d$	$s_{23}\ u_l^{19} u_b^{6} o_r^{19} w_r p_d$	$s_{24}\ u_l^{55} u_b^{37} o_r^{19} i^{54} p_b p_d$		
13		$s_{25}\ w_l w_b w_r p_d$	$s_{25}\ w_l w_b w_r p_b$	$s_{25}\ w_l w_b w_r p_b p_d$	$s_{25}\ u_l^{55} u_b^{77} u_r^{113} p_a p_d$	$s_{25}\ u_l^{72} u_b^{59} u_r^{95} p_b p_d$	$s_{25}\ u_l^{55} u_b^{37} u_r^{77} p_b p_d$
			$s_{25}\ w_l w_b w_r p_d$	$s_{25}\ u_l^{19} u_b^{6} w_r p_d$	$s_{25}\ u_l^{55} u_b^{37} u_r^{72} p_b p_d$		
14		$s_7\ w_l w_b w_r p_d$	$s_7\ w_l w_b w_r p_b$	$s_7\ w_l w_b w_r p_b p_d$	$s_7\ u_l^{55} u_b^{77} u_r^{113} p_a p_d$	$s_7\ u_l^{72} u_b^{59} u_r^{95} p_b p_d$	$s_7\ u_l^{55} u_b^{37} u_r^{77} p_b p_d$
			$s_7\ w_l w_b w_r p_d$	$s_7\ u_l^{19} u_b^{6} w_r p_d$	$s_7\ u_l^{55} u_b^{37} u_r^{72} p_b p_d$		
15		$s_{26}\ w_l w_b w_r p_d$	$s_{26}\ w_l w_b w_r p_b$	$s_{26}\ w_l w_b w_r p_b p_d$	$s_{26}\ u_l^{55} u_b^{77} u_r^{113} p_a p_d$	$s_{26}\ u_l^{72} u_b^{59} u_r^{95} p_b p_d$	$s_{26}\ u_l^{55} u_b^{37} u_r^{77} p_b p_d$
			$s_{26}\ w_l w_b w_r p_d$	$s_{26}\ u_l^{19} u_b^{6} w_r p_d$	$s_{26}\ u_l^{55} u_b^{37} u_r^{72} p_b p_d$		

[10] Păun, G., Pérez-Jiménez, M.J.: Solving problems in a distributed way in membrane computing: dP systems. International Journal of Computers, Communications and Control 5(2), 238–252 (2010)