ELSEVIER

Artificial Intelligence 91 (1997) 3-32

Artificial Intelligence

# Grammatical description of behaviors of ordinary differential equations in two-dimensional phase space

Toyoaki Nishida [1]

*Graduate School of Information Science, Nara Institute of Science and Technology, 8916-5 Takayama, Ikoma, Nara 630-01, Japan*

Received April 1996

**Abstract**

The major task of qualitative analysis of systems of ordinary differential equations is to recognize the global pattern of solution curves in the phase space. In this paper, I present a *flow grammar*, a grammatical specification of all possible patterns of solution curves one may see in the phase space.

I describe a *flow pattern*, a semi-symbolic representation of the patterns of solution patterns in the phase space, and show how an important class of flow patterns can be specified by the flow grammar. I then show that the flow grammar presented in this paper can generate any flow pattern resulting from any structurally stable flow on a plane. I also describe several properties of the flow grammar related to the enumeration of patterns. In particular, I estimate the upper limit of the number of applications of rewriting rules needed to derive a given flow pattern. Finally, I describe how the flow grammar is used in qualitative analysis to plan, monitor, and interpret the result of numerical computation. © 1997 Elsevier Science B.V.

*Keywords:* Qualitative reasoning; Intelligent scientific computation; Dynamical systems theory; Flow grammar; Flow mapping

## 1. Introduction

In many domains in science and technology, one is often faced with understanding the behavior of differential equations. The behavior of differential equations becomes fairly

[1] E-mail: nishida@is.aist-nara.ac.jp.

*PII* S0004-3702(96)00055-0

complex and subtle when nonlinearity comes into play. Even a small numerical error may result in an entirely different conclusion, due to sensitivity to initial conditions.

In applied mathematics [2,3], it is known that geometric methods are effective in qualitative analysis of nonlinear differential equations. Thus, one can understand qualitative behaviors of ordinary differential equations (ODEs) by analyzing geometric and topological features of solution curves in the phase space.

I have taken a geometric approach and developed a program called PSX2NL [6,7]. Given a system of ODEs, PSX2NL takes the following three steps to produce a semi-symbolic description of the behavior:

- *Step 1.* Collect geometric features of solution curves using varieties of quantitative techniques.
- *Step 2.* Infer the topology of the phase portrait from geometric cues.
- *Step 3.* Reason about the global behavior by analyzing the topology of the phase portrait.

In this paper, I present a *flow grammar*, a grammatical specification of all possible patterns of solution curves one may see in the phase space. I describe *flow pattern*, a semi-symbolic representation of the patterns of solution patterns in the phase space, and show how an important class of flow patterns can be specified by the flow grammar. I then show that the flow grammar presented in this paper can generate any flow pattern resulting from any structurally stable flow on a plane. I also describe several properties of the flow grammar related to the enumeration of patterns. In particular, I estimate the upper limit of the number of applications of rewriting rules needed to derive a given flow pattern. Finally, I describe how the flow grammar is used in qualitative analysis to plan, monitor, and interpret the result of numerical computation.

## 2. Qualitative theory of ODEs and qualitative analysis

In this paper, we consider systems of ODEs with two state variables:

$$\frac{\mathrm{d}\boldsymbol{x}}{\mathrm{d}t} = f(\boldsymbol{x}), \tag{1}$$

where

$$\boldsymbol{x}(t) = \{x_1(t), x_2(t)\} \quad (t \in \mathbb{R}), \qquad f : \mathbb{R}^2 \to \mathbb{R}^2.$$

The space $\mathbb{R}^2$ spanned by the two state variables in (1) is called the *phase space*. Eq. (1) is said to introduce the *vector field* into the phase space, as the right-hand side of Eq. (1) specifies the velocity of state change at each point in the phase space. A trajectory resulting from traversing the phase portrait according to the vector field is called a *solution curve*, or an *orbit*. Each orbit represents a solution under some initial condition, and may be characterized as a function $\phi_{\boldsymbol{x}}(t) : \mathbb{R} \to \mathbb{R}^2$ which returns a point in the phase space as a function of time. The collection of all orbits in the phase space is called the *phase portrait*, which might be thought of as introducing a *flow* characterized as a function $\Phi_t(\boldsymbol{x}) : \boldsymbol{x} \in \mathbb{R}^2 \mapsto \phi_{\boldsymbol{x}}(t) \in \mathbb{R}^2$.

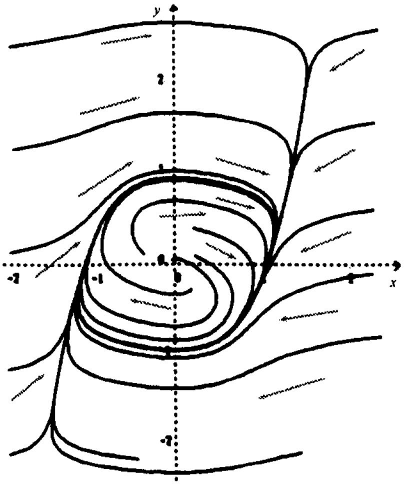


Fig. 1. The phase portrait for Eq. (2).

The following is a typical system of ODEs we investigate in this paper:

$$\begin{cases} \dfrac{\mathrm{d}x}{\mathrm{d}t} = -2x^3 + 2x + 2y, \\ \dfrac{\mathrm{d}y}{\mathrm{d}t} = -x. \end{cases} \tag{2}$$

This is nonlinear in the sense that the right-hand sides contain nonlinear terms such as $x^3$. Unlike linear differential equations, no general algorithm is known for solving nonlinear ODEs analytically. In addition, the behavior of nonlinear differential equations may become fairly complex in certain situations. According to dynamical systems theory [2, 3], studying geometric and topological properties of the phase portrait provides a powerful means of understanding qualitative behaviors of ODEs even when the explicit form of solution is not available.

Fig. 1 illustrates a portion of the phase portrait of Eq. (2), where a number of orbits are approximately drawn by numeric integration.

Orbits are classified according to their asymptotic behaviors as $t \to \pm\infty$. As for systems of ODEs with two state variables, it is proved that orbits may either diverge to place at infinity, or tend towards a *fixed point* (an orbit consisting of a single point which makes the right-hand side of Eq. (1) *zero*) or a *limit cycle* (a cyclic orbit which attracts or repels nearby orbits).[2]

As for the phase portrait in Fig. 1, one may conjecture the existence of a fixed point at $(0, 0)$ and a limit cycle around it. It is known that all orbits except the fixed point at $(0, 0)$ tend towards the limit cycle as $t \to \infty$ and exhibit a periodical behavior in a long run. In this sense, the limit cycle is called an $\omega$-limit cycle. In contrast, those limit cycles which attract nearby orbits as $t \to -\infty$ are called $\alpha$-limit cycles.

[2] This follows from the Poincaré–Bendixson theorem. See [3, p. 248] for more details.

The goal of qualitative analysis is to identify the patterns of behaviors under all initial conditions, by investigating geometric and topological properties of orbits of a given system of ODEs. We do not have a general means for representing $\boldsymbol{x}$ as an explicit function of $t$ by symbolically manipulating Eq. (1). It is possible to obtain an approximate picture of an orbit by using a numerical integration algorithm such as the Runge–Kutta algorithm, as was done for examples in this paper. However, numerical methods support only a small portion of the entire process of understanding the behavior, as pointed out in [11]. Moreover, numerical methods may result in logical errors or incompleteness caused by numeric errors and incompleteness.

In order to supplement numerical methods, I propose a method of representing knowledge about patterns of orbits in the phase space and applying it to planning, monitoring and interpreting numeric computation. The basic idea is to partition the phase space into regions called cells, and characterize the fragments of orbits in each region as *flow mappings*, and put together flow mappings for these regions to reason about global behavior. Currently, our technique cannot handle flow in open regions. This means that our method of global analysis is limited to flow in a bounded region.

In the next two sections, I first describe a semi-symbolic representation of patterns of orbits in a bounded region and then introduce a grammatical specification of possible patterns of orbits in a bounded region.

## 3. Semi-symbolic description of patterns of orbits in a cell

We assume a cell to be a convex region and the phase space to be two-dimensional. Generally, a fragment of an orbit in a cell $C$ either approaches a fixed point or a limit cycle, or cuts across a boundary $\partial C$ of $C$, as $t \to \pm\infty$.

### 3.1. Bundle of orbit intervals

We call a contiguous segment of an orbit an *orbit interval* and denote it as $\phi(t)$, where $t \in \mathbb{R}$. When either end of the orbit interval $\phi$ is a fixed point $f$, the orbit interval does not contain $f$ although $\phi(t)$ asymptotically approaches $f$ as $t \to \infty$ or $-\infty$. To take into account an extended orbit interval containing both ends, we extend the region of parameters from the set of real numbers $\mathbb{R}$ to the set of hyperreals $\mathbb{R}^* \equiv \mathbb{R} \cup \{-\infty, \infty\}$ and we consider the closure of the orbit interval $\phi$, denoted as $\phi^*$.

Given a cell $C$ and a couple of orbit intervals $\phi$ and $\psi$ contained in $C$, we define the *distance* $d(\phi, \psi)$ between $\phi$ and $\psi$ as the maximal value of the minimal distance between a point on one orbit interval and another orbit interval. Namely,

$$d(\phi, \psi) = r$$

iff

$$\exists u, v \in \mathbb{R}^* [\phi^*(u) \in C \wedge \psi^*(v) \in C \wedge |\phi^*(u) - \psi^*(v)| = r],$$

$$\forall u \in \mathbb{R}^* [\phi^*(u) \in C \to \exists v \in \mathbb{R}^* \psi^*(v) \in C \wedge |\phi^*(u) - \psi^*(v)| \leqslant r],$$

$$\forall v \in \mathbb{R}^* [\psi^*(v) \in C \to \exists u \in \mathbb{R}^* \phi^*(u) \in C \wedge |\phi^*(u) - \psi^*(v)| \leqslant r].$$

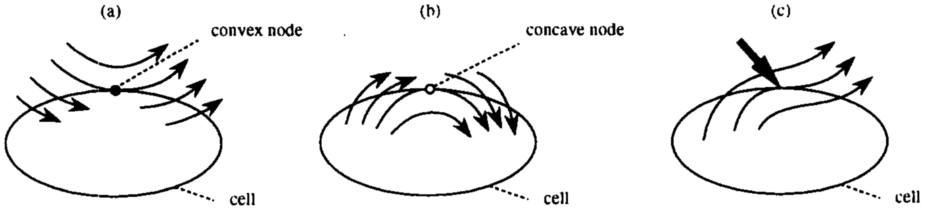


Fig. 2. Points of contact.

A *bundle of orbit intervals* in a cell is an aggregation of nearby orbit intervals in a cell. Namely, given a cell $C$ and orbit intervals $\phi$ and $\psi$ in $C$, we consider that $\phi$ and $\psi$ belong to the same bundle of orbit intervals iff for any positive number $\varepsilon$ there exists orbit intervals $\gamma_1, \ldots, \gamma_m$ such that

$$d(\phi, \gamma_1) < \varepsilon \wedge \forall i (1 \leqslant i \leqslant m-1)\ [d(\gamma_i, \gamma_{i+1}) < \varepsilon] \wedge d(\gamma_m, \psi) < \varepsilon.$$

The collection of orbit intervals in a cell is classified into the collection of bundles of orbit intervals. It is noted that the set of bundles of orbit intervals is determined by the geometric relations between the flow and a cell.

In order to compute the collection of orbit intervals, we identify the following characteristic points for a given cell:

- Fixed points in the cell: the type of a fixed point is either a sink, a source, or a saddle node.
- Points of contact on the boundary of the cell where an orbit is tangent to the boundary of the cell and lies in the same side of the boundary immediately before and after contact. A point of contact is called a *concave node* if the orbit passing on it lies inside the region immediately before and after contact, as shown in Fig. 2(a). Otherwise a point of contact is called a *convex node* (Fig. 2(b)). Those points such as the one in Fig. 2(c) are not called a point of contact, for the orbit on the point lies in the different side of the cell.
- *Landmarks*: points where an orbit running on a concave node cuts across the cell boundary, or a stable or unstable manifold of a saddle node cuts across the cell boundary.

We can compute the location of fixed points of Eq. (1) by solving

$$f(\boldsymbol{x}) = 0,$$

and that of points of contact by solving

$$f(\boldsymbol{x}) \cdot \boldsymbol{a} = 0,$$

where $\boldsymbol{a}$ denotes the orientation orthogonal to the boundary edge. In order to compute the location of landmarks, we have to track orbits from saddle nodes and concave nodes to find the position where they cut across the cell boundary. We use numeric integration for tracking orbits.

A contiguous segment of the cell boundary delimited by points of contact is called a *boundary segment*. At one boundary segment, the orientation of the flow is qualitatively

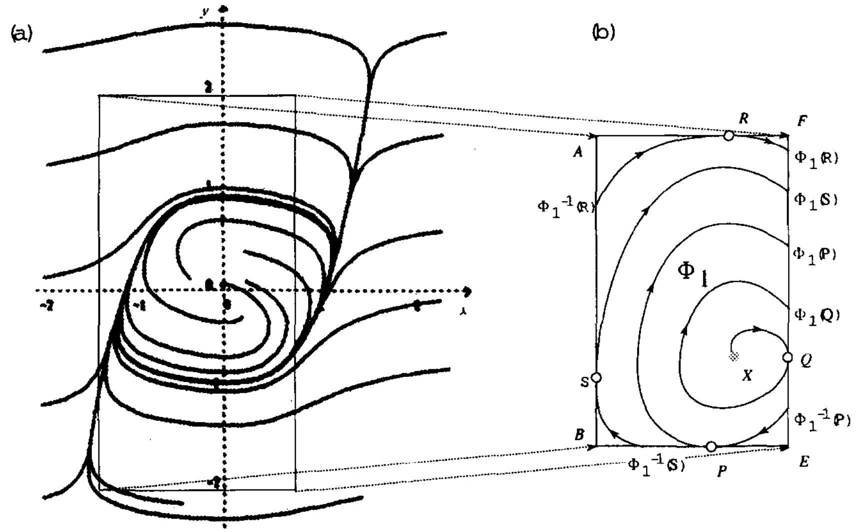


Fig. 3. (a) Part of the flow in Fig. 1. (b) Its schematization.

the same, equally coming in or going out the cell. A boundary segment is called an *entrance segment* and *exit segment* if the flow is coming in and going out the cell, respectively. A boundary segment may be further partitioned into one or more *boundary edges* by landmarks on the boundary segment. A boundary segment which does not contain any landmarks makes one boundary edge. The collection of orbit intervals makes a bundle of orbit intervals if every orbit interval cuts across the same boundary edge.

*Example*

Let us consider part of the flow in Fig. 1, as in Fig. 3(a). The flow there can be schematized as Fig. 3(b). The cell contains four concave nodes and four convex nodes, which partition the boundary of the cell into eight boundary segments

$$\{\overline{A,S},\ \overline{S,B},\ \overline{B,P},\ \ldots,\ \overline{R,A}\}.$$

Those boundary segments are further partitioned into boundary edges. For example, boundary segment $\overline{A,S}$ is partitioned into boundary edges $\overline{A,\Phi_1^{-1}(R)}$ and $\overline{\Phi_1^{-1}(R),S}$ by landmark $\Phi_1^{-1}(R)$, where $\Phi_1$ stands for the flow in the cell and $\Phi_1^{-1}(R)$ for the inverse image of the concave node $R$.

### 3.2. Flow mappings

We represent a flow as a set of *flow mappings* which specifies how the flow maps point in the phase space. For example, consider the flow in the region *ABEF* shown in Fig. 3(a). Some points on segment $\overline{E,F}$ are mapped from the fixed point $X$ in the region, while other come from other points on the boundary. Although all points on the open segment $\overline{\Phi_1(Q),F}$ come from other points on the boundary, the source is not one continuous segment, but a sum of four open segments: $\overline{\Phi_1^{-1}(P),Q}$, $\overline{\Phi_1^{-1}(S),P}$, $\overline{\Phi_1^{-1}(R),S}$, and $\overline{F,R}$, and three landmarks delimiting them: $\Phi_1(P)$, $\Phi_1(S)$, and $\Phi_1(R)$, so we represent the flow related to $\overline{\Phi_1(Q),F}$ as a collection of seven flow mappings:

$$\begin{aligned}
&\overline{\Phi_1^{-1}(P),Q} \longrightarrow \overline{\Phi_1(Q),\Phi_1(P)}\\
&\oplus \overline{\Phi_1^{-1}(S),P} \longrightarrow \overline{\Phi_1(P),\Phi_1(S)}\\
&\oplus \overline{\Phi_1^{-1}(R),S} \longrightarrow \overline{\Phi_1(S),\Phi_1(R)}\\
&\oplus \overline{F,R} \longrightarrow \overline{\Phi_1(R),F}\\
&\oplus P \longrightarrow \Phi_1(P)\\
&\oplus S \longrightarrow \Phi_1(S)\\
&\oplus R \longrightarrow \Phi_1(R).
\end{aligned}$$

Thus, the flow in $ABEF$ is represented as a sum of sixteen flow mappings, as follows:[3]

$$\begin{aligned}
\Phi_1 = \Phi_{1,1} &: \overline{A,\Phi_1^{-1}(R)} \longrightarrow \overline{R,A}\\
&\oplus \Phi_{1,2} : \overline{\Phi_1^{-1}(R),S} \longrightarrow \overline{\Phi_1(S),\Phi_1(R)}\\
&\oplus \Phi_{1,3} : \overline{B,\Phi_1^{-1}(S)} \longrightarrow \overline{S,B}\\
&\oplus \Phi_{1,4} : \overline{\Phi_1^{-1}(S),P} \longrightarrow \overline{\Phi_1(P),\Phi_1(S)}\\
&\oplus \Phi_{1,5} : \overline{E,\Phi_1^{-1}(P)} \longrightarrow \overline{P,E}\\
&\oplus \Phi_{1,6} : \overline{\Phi_1^{-1}(P),Q} \longrightarrow \overline{\Phi_1(Q),\Phi_1(P)}\\
&\oplus \Phi_{1,7} : X \longrightarrow \overline{Q,\Phi_1(Q)}\\
&\oplus \Phi_{1,8} : \overline{F,R} \longrightarrow \overline{\Phi_1(R),F}\\
&\oplus \Phi_1^{-1}(R) \longrightarrow R \oplus R \longrightarrow \Phi_1(R)\\
&\oplus \Phi_1^{-1}(S) \longrightarrow S \oplus S \longrightarrow \Phi_1(S)\\
&\oplus \Phi_1^{-1}(P) \longrightarrow P \oplus X \longrightarrow Q\\
&\oplus Q \longrightarrow \Phi_1(Q) \oplus X.
\end{aligned} \tag{3}$$

Generally, we represent a bundle of orbit intervals $\Phi$ in a cell $C$ as a *flow mapping*

$$\Phi : \alpha \longrightarrow \beta.$$

$\alpha$ and $\beta$ are called *generalized source* and *generalized sink*, respectively. They are either a boundary edge or a fixed point or a limit cycle which orbits in $\Phi$ cut across or tend towards when $t \longrightarrow -\infty$ or $t \longrightarrow \infty$.

[3] The first eight are essential. The others involve some subtlety, but this is not critical to our discussion.

(a) Symbolic representation

| | |
|---|---|
| constituents: | $X$ : type = fixed-point, source<br>$A$ : type = convex-node<br>$\overline{A,\ \Phi_1^{-1}(R)}$ : type = boundary-edge, entrance<br>$\Phi_1^{-1}(R)$ : type = landmark, entrance<br>$\overline{\Phi_1^{-1}(R),\ S}$ : type = boundary-edge, entrance<br>... |
| boundary-list: | $[A, \overline{A,\ \Phi_1^{-1}(R)}, \Phi_1^{-1}(R), \overline{\Phi_1^{-1}(R),\ S}, \ldots, \overline{R,\ A}]$ |
| flow-mappings: | $\Phi_{1,1} : \overline{A,\ \Phi_1^{-1}(R)} \to \overline{R,\ A}$<br>...<br>$\Phi_{1,1,2} : \Phi_1^{-1}(R) \to R$<br>...<br>$\Phi_{1,0} : X \to X$ |

(b) Schematic diagram

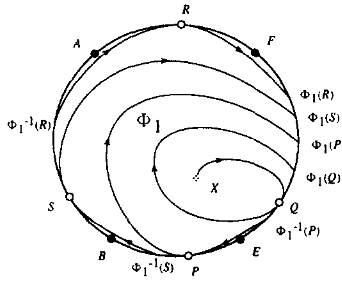


Fig. 4. The flow pattern for the local flow in region *ABEF* in Fig. 3(b).

### 3.3. Flow patterns

We use *flow patterns* to represent the collection of orbit intervals in a cell $C$ as a collection of flow mappings. A flow pattern consists of:

- *constituents*: the set of geometric constituents of $C$ such as fixed points, limit cycles, convex nodes, concave nodes, boundary edges, landmarks, and their geometric properties;
- *boundary-list*: the counterclockwise order of geometric cues on the cell boundary;
- *flow-mappings*: the set of flow mappings characterizing the flow in $C$.

Fig. 4 shows the flow pattern for the local flow in region $ABEF$ in Fig. 3(b). A flow pattern focuses on the topology, rather than the shape, of the flow in a given cell. Note that there may exist multiple flow patterns for the same flow in the same cell, which differ from each other only in labeling. Flow patterns $x$ and $y$ are said to be *equivalent* if they differ only in the way labels are attached.

The computability and complexity of generating flow patterns depend on the class of ODEs. For two-dimensional piecewise linear differential equations, which result from approximating each occurrence of nonlinear terms in a nonlinear differential equation by a set of connected pieces of linear functions, almost all of the process is computable [8]. For more complex classes of ODEs, difficulties arise mainly because complete information may not be available due to the complexity of mathematical problems encountered [7]. This implies that the procedure may fail when it tries to divide the phase space into uniform regions or to annotate flow at the boundary of the regions. Moreover, handcrafting procedures which can deal with these situations would be quite painstaking because of unmanageably many combinations of possibilities. To overcome these difficulties, we take a grammatical approach, which will be presented in the next section.

The main stream of global analysis is to merge flow patterns in turn and examine topological properties of resulting flow patterns for larger regions. Limit cycles, if any, can be detected in finite steps as far as it is not totally contained in a single region as a result of phase space partition and the local flows in related regions are properly analyzed [7]. We use several heuristics to back up the incompleteness. Since attracting and repelling sets of planar ODEs are either fixed points or limit cycles, and since fixed points are determined in analysis of local flow, the above gives a complete process for global analysis in theory. Of course, there is a chance that the above method may fail or produce an incorrect result, due to the failure in characterizing the local flow or numerical error. This is a common difficulty we encounter in addressing nonlinear problems.

### 3.4. *Composition of flow mappings and recursive mapping*

Given a couple of bundles of flow mappings $\Phi_1 : I \rightarrow J$ and $\Phi_2 : J \rightarrow K$, let us define *composition* $\Phi_2 \circ \Phi_1$ of $\Phi_1$ and $\Phi_2$ as:

$$\Phi_2 \circ \Phi_1(x) = \Phi_2(\Phi_1(x)) = y \quad \text{iff} \quad \exists z\ [\Phi_1(x) = z,\ \Phi_2(z) = y].$$

We can grasp the behavioral characteristics of the solution of ODEs by partitioning the phase space into cells, and investigating the topological structure of compositions of flow mappings for each cell. We call a composite flow mapping $\phi_m \circ \cdots \circ \phi_1$ a *contracting recursive mapping*, if the range is a subset of the domain, namely,

$$\phi_m \circ \cdots \circ \phi_1(I) \subset I.$$

Similarly, we call $\phi_m \circ \cdots \circ \phi_1$ an *extending recursive mapping*, if

$$\phi_m \circ \cdots \circ \phi_1(I) \supset I.$$

Existence of a contracting recursive mapping entails existence of an attractor. Similarly, existence of an extending recursive mapping entails existence of a repellor.

## 4. Flow grammar

A *flow grammar* specifies all possible flow patterns in closed regions as a formal grammar.[4] Generally, a flow grammar is a system $G = \langle B, D, C \rangle$, where $B$ is a set of *basic flow patterns*, $D$ a set of *distortion rules*, and $C$ a set of *composition rules*. We say that a flow pattern is *generated* by $G = \langle B, D, C \rangle$ if it is obtained by applying the finite number of rules in $D$ and $C$ to a flow pattern in $B$.

### *4.1. Flow grammar FG*1

In the remainder of this paper, we will describe one particular flow grammar called *FG*1, though other formulations might be possible as well. Particularly, we assume that closed regions do not have fixed points on the boundary and that the given flow is *structurally stable.*[5] Moreover, the flow grammar presented below does not generate limit cycles for efficiency. Limit cycles are handled in the algorithm which calls for the flow grammar [7].

*4.1.1. Basic flow patterns*

Basic flow patterns specify flow patterns which contain at most one fixed point. We have four basic flow patterns as illustrated in Fig. 5. Basic flow patterns have at most one type of points of contact on the boundary. The flow pattern in Fig. 5(a) is the simplest among those not containing any fixed points. Flow patterns in Fig. 5(b)-(d) are the simplest among those containing exactly one fixed point.

*4.1.2. Distortion rules*

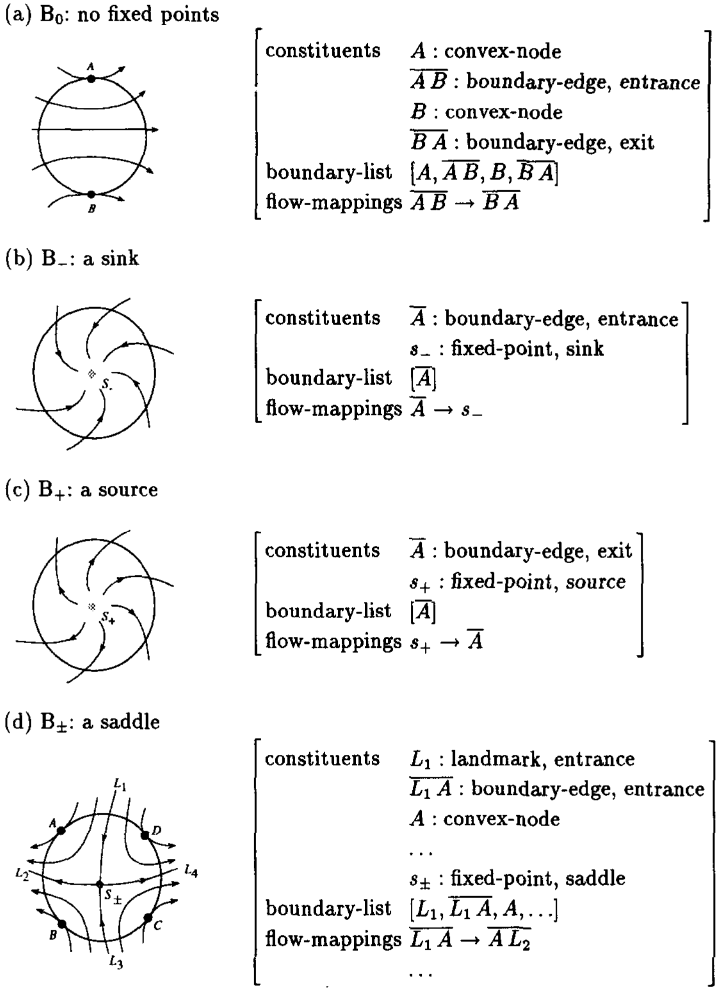
Distortion rules specify flow patterns arising when the relative geometric relation between the flow and the boundary segment becomes complex, without increasing the number of fixed points in the cell. We have two distortion rules as shown in Fig. 6(a) and (b). One introduces a sequence of a concave node and a convex node into a boundary segment and the other introduces them in reverse order.

*4.1.3. Composition rules*

Composition rules specify the way in which flow patterns with more than one fixed point are computed. We consider a flow pattern with $n$ ($n \geqslant 2$) fixed points to be generated by fusing a couple of flow patterns: one with $(n-1)$ fixed points and another

[4] The notion of flow grammar was inspired by process grammar [5].

[5] Structurally stable flows are those which persist under an infinitesimal perturbation to their parameters. If the purpose is to analyze ODEs for physical systems, only structurally stable systems may be observed. Peixoto's theorem suggests that fixed points which may appear in structurally stable flows are either *sinks*, *sources*, or *saddle points* (see [2, p. 60] for more detail).



(a) $B_0$: no fixed points

| | |
|---|---|
| constituents | $A$ : convex-node |
| | $\overline{AB}$ : boundary-edge, entrance |
| | $B$ : convex-node |
| | $\overline{BA}$ : boundary-edge, exit |
| boundary-list | $[A, \overline{AB}, B, \overline{BA}]$ |
| flow-mappings | $\overline{AB} \to \overline{BA}$ |

(b) $B_-$: a sink

| | |
|---|---|
| constituents | $\overline{A}$ : boundary-edge, entrance |
| | $s_-$ : fixed-point, sink |
| boundary-list | $[\overline{A}]$ |
| flow-mappings | $\overline{A} \to s_-$ |

(c) $B_+$: a source

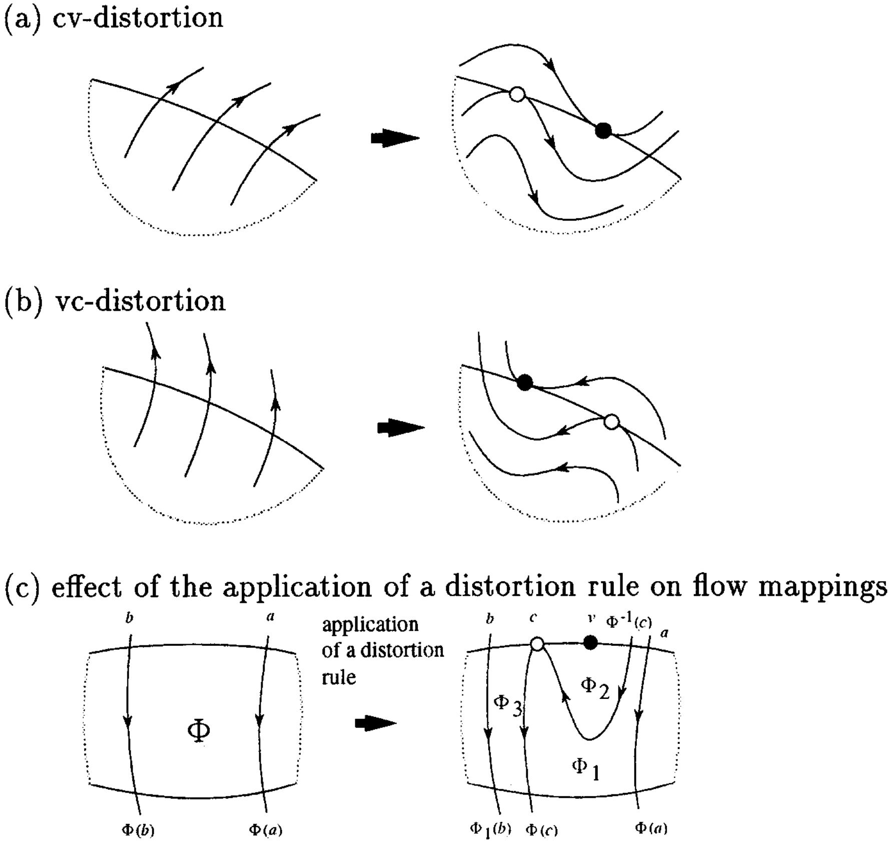
| | |
|---|---|
| constituents | $\overline{A}$ : boundary-edge, exit |
| | $s_+$ : fixed-point, source |
| boundary-list | $[\overline{A}]$ |
| flow-mappings | $s_+ \to \overline{A}$ |

(d) $B_\pm$: a saddle

| | |
|---|---|
| constituents | $L_1$ : landmark, entrance |
| | $\overline{L_1 A}$ : boundary-edge, entrance |
| | $A$ : convex-node |
| | ... |
| | $s_\pm$ : fixed-point, saddle |
| boundary-list | $[L_1, \overline{L_1 A}, A, \ldots]$ |
| flow-mappings | $\overline{L_1 A} \to \overline{A L_2}$ |
| | ... |

Fig. 5. Basic flow patterns.

with exactly one fixed point, as shown in Fig. 7 where two flow patterns $P_1$ and $P_2$ are fused together at boundary segments $\overline{a_1 b_1}$ of $P_1$ and $\overline{a_2 b_2}$ of $P_2$. In order for flow patterns $P_1$ and $P_2$ to be properly fused, the flow at the boundary segments $(a_1, b_1)$ and $(a_2, b_2)$ should be complementary: outgoing (incoming) flow of $P_1$ should correspond to incoming (outgoing) flow of $P_2$, and each convex (concave) node of $P_1$ should correspond to a convex (concave) node of $P_2$. See Fig. 8. At the end points $a_i$ and $b_i$, flow should be either one of the four patterns shown in Fig. 9. Note that we assume that each cell has to contain at least one fixed point in the cell, for all flow patterns resulting from fusing a flow pattern $P$ and another not containing any fixed points can also be generated by applying distortion rules to $P$.



$$\Phi : \overline{a,b} \longrightarrow \overline{\phi(b),\phi(a)}$$

$$\Phi_1 : \overline{a,\Phi^{-1}(c)} \longrightarrow \overline{\phi(c),\phi(a)}$$

$$\oplus\ \Phi_2 : \overline{\Phi^{-1}(c),v} \longrightarrow \overline{v,c}$$

$$\oplus\ \Phi_3 : \overline{c,b} \longrightarrow \overline{\phi(b),\phi(c)}$$

Fig. 6. Distortion rules.

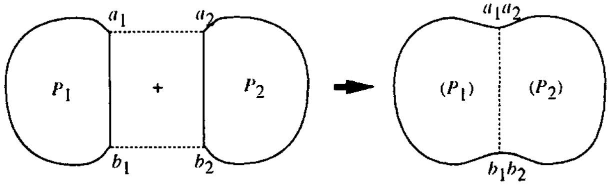


Fig. 7. Composition of flow patterns by fusing.

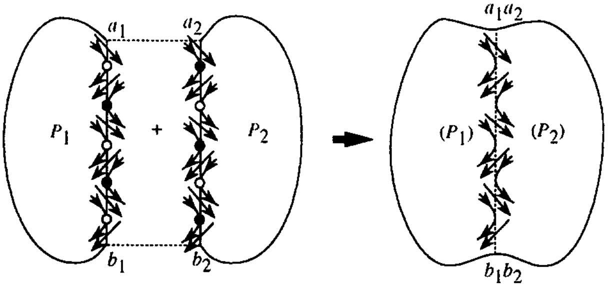


Fig. 8. Conditions for "fusing" a couple of flow patterns.

Fig. 10 illustrates how the flow pattern shown in Fig. 4 can be obtained by applying a sequence of distortion rules to the basic flow pattern $B_+$. Fig. 11 illustrates flow patterns formed by applying the composition rule to basic flow patterns $B_\pm$ and $B_-$.

$FG1$ introduced so far has the following characteristics:

**Proposition 1.** *Any flow pattern representing a structurally stable flow in a region on the two-dimensional plane can be generated by* $FG1$ *unless the flow contains a limit cycle.*

**Proof.** Peixoto's theorem[6] endorses that the simplest flow patterns generated by any $C^1$ flow on the two-dimensional plane and containing at most one fixed point are only those shown in Fig. 5.

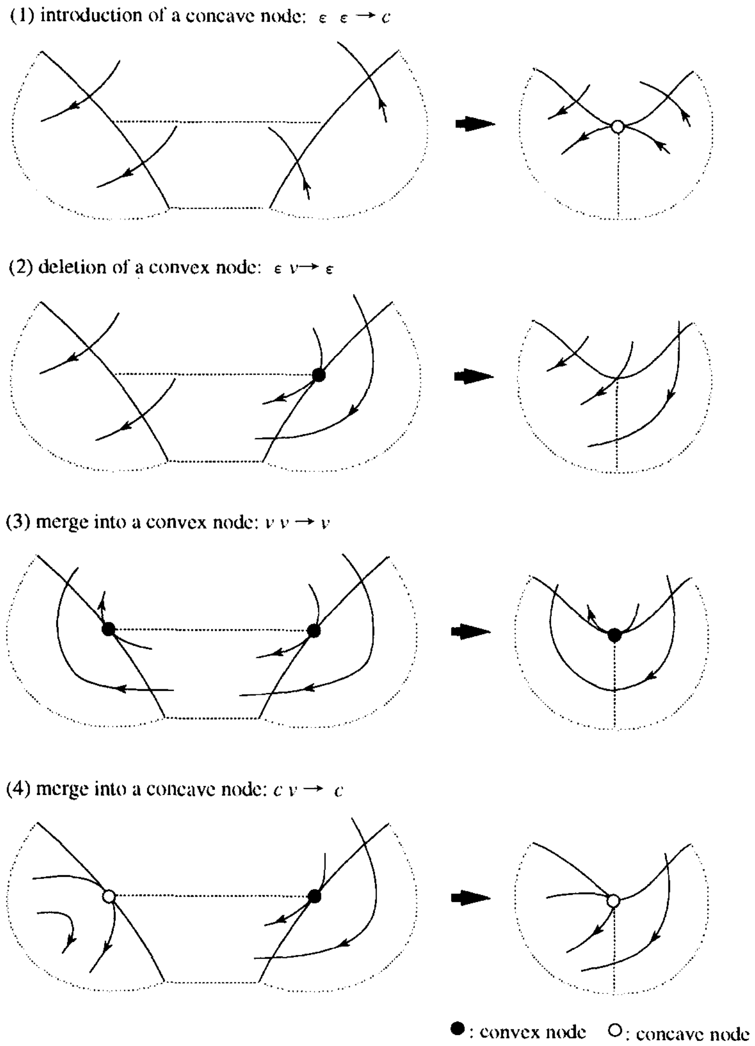
Now suppose that we are given an arbitrary structurally stable flow in a region on the two-dimensional plane not containing any limit cycle, and consider the process of growing and fusing small cells around fixed points in the region until they become as large as the region.[7] It is possible to find a small region around a fixed point whose flow pattern conforms to one of the basic flow patterns shown in Fig. 5, for the microscopic nature of a $C^1$ flow is relatively simple in the sense that it can be approximated by a linear flow in an appropriately small region.

The distortion rules shown in Fig. 6 cover all situations where convex and concave nodes are introduced due to the complication of the geometric relation between the cell boundary and the flow during growing single cells. The composition rule shown in Fig. 7 covers all situations where the flow is linked as two cells are put together. □

We might have designed $FG1$ so that each flow pattern could contain one or more limit cycles. However, such a flow grammar would not be useful for qualitative analysis, for such a design would unmanageably increase the number of possible flow patterns for the same observation. Our program PSX2NL [6,7] has a mechanism of partitioning

[6] See [2, p. 60] for more detail.

[7] When the region contains no fixed points, we will start by a region whose flow pattern conforms to the one shown in Fig. 5(a).



(1) introduction of a concave node: $\varepsilon\ \varepsilon \rightarrow c$

(2) deletion of a convex node: $\varepsilon\ v \rightarrow \varepsilon$

(3) merge into a convex node: $v\ v \rightarrow v$

(4) merge into a concave node: $c\ v \rightarrow c$

● : convex node ○ : concave node

Fig. 9. Four patterns governing fusion of flow at the end points.

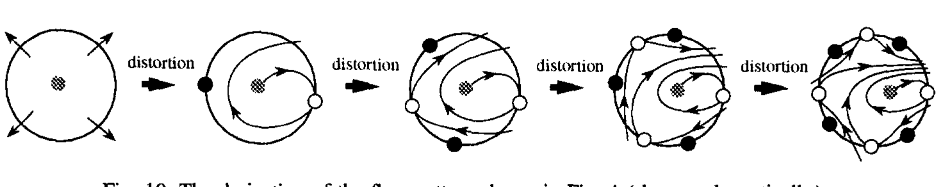


Fig. 10. The derivation of the flow pattern shown in Fig. 4 (drawn schematically)

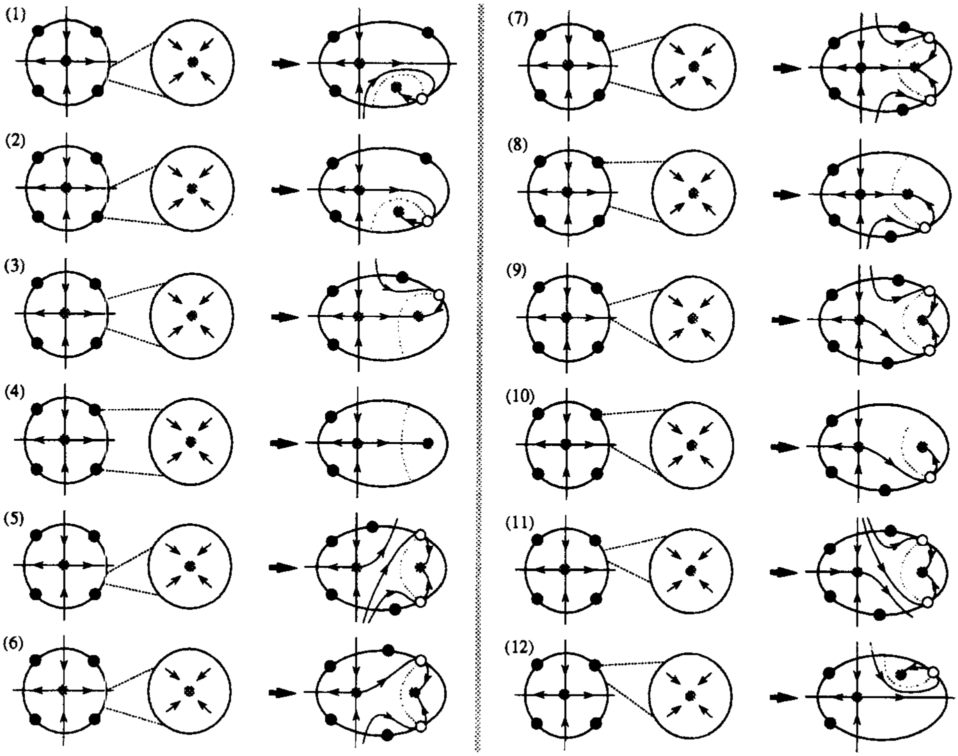


Fig. 11. Flow patterns formed from basic flow patterns $B_{\pm}$ and $B_{-}$

a cell into smaller pieces so that no limit cycles may not be contained in each piece, if it finds out the symptom of limit cycles.

### 4.2. Indexing flow patterns

We classify flow patterns by the number of fixed points and points of contact involved. We le called *flow index* as an index to the set of flow patterns. A flow index is a $, n_c, n_{sa}, n_{ss}\rangle$, where $n_v$ denotes the number of convex nodes in the flow patt at of concave nodes, $n_{sa}$ that of saddle nodes, and $n_{ss}$ that of sinks or sources.

For example, the flow index for the basic flow pattern $B_0$ is $\langle 2,0,0,0\rangle$, that for $B_{-}$ and $B_{-}$ is both $\langle 0,0,0,1\rangle$, and that for $B_{\pm}$ is $\langle 4,0,1,0\rangle$.

As for the flow patterns in Fig. 11, the flow index for the flow pattern (4) is $\langle 2,0,1,1\rangle$; that for (1), (2), (3), (8), (10), and (12) is equally $\langle 3,1,1,1\rangle$; and that for (5), (6), (7), (9), and (11) is $\langle 4,2,1,1\rangle$.

Although the index appears four-dimensional, it is actually three-dimensional, as shown in the following property:

**Proposition 2.** *For any $C^1$ structurally stable flow and a closed region $R$, let the number of saddle points and sinks or sources in $R$ be $n_{\mathrm{sa}}$ and $n_{\mathrm{ss}}$, respectively. Let the number of convex nodes and concave nodes at the boundary $\partial R$ of $R$ be $n_{\mathrm{v}}$ and $n_{\mathrm{c}}$, respectively. Then we have the following relation:*

$$n_{\mathrm{v}} - n_{\mathrm{c}} = 2 \times (n_{\mathrm{sa}} - n_{\mathrm{ss}} + 1). \tag{4}$$

**Proof.** We use mathematical induction to prove Eq. (4). The idea is to observe that the process of partitioning the $C^1$ flow in a given cell is exactly the inverse of the process of generating the associated flow pattern by applying distortion and composition rules to the basic flow patterns.

First, Eq. (4) holds for basic flow patterns.[8]

Second, the application of a distortion rule preserves Eq. (4), for it will increase the number of concave and convex nodes by one, respectively.

Third, consider now that we apply the composition rule to fuse two flow patterns with flow indices $\langle n_{\mathrm{v},1}, n_{\mathrm{c},1}, n_{\mathrm{sa},1}, n_{\mathrm{ss},1}\rangle$ and $\langle n_{\mathrm{v},2}, n_{\mathrm{c},2}, n_{\mathrm{sa},2}, n_{\mathrm{ss},2}\rangle$, and assume that Eq. (4) holds for each of them. Thus, we have

$$n_{\mathrm{v},1} - n_{\mathrm{c},1} = 2 \times (n_{\mathrm{sa},1} - n_{\mathrm{ss},1} + 1), \tag{5}$$

$$n_{\mathrm{v},2} - n_{\mathrm{c},2} = 2 \times (n_{\mathrm{sa},2} - n_{\mathrm{ss},2} + 1). \tag{6}$$

On the other hand, let $\langle n_{\mathrm{v}}, n_{\mathrm{c}}, n_{\mathrm{sa}}, n_{\mathrm{ss}}\rangle$ be the flow index for the flow pattern resulting from the two flow patterns. Then, we have

$$n_{\mathrm{sa}} = n_{\mathrm{sa},1} + n_{\mathrm{sa},2}, \tag{7}$$

$$n_{\mathrm{ss}} = n_{\mathrm{ss},1} + n_{\mathrm{ss},2}. \tag{8}$$

The same number of convex nodes and concave nodes disappears at the segments for fusion except the two end points, as shown in Fig. 8. A new convex or concave node may be introduced at the two ends of the segment for fusion, as shown in Fig. 9. In each of the four cases, $n_{\mathrm{v}} - n_{\mathrm{c}}$ decreases by one. Hence, we have

$$n_{\mathrm{c}} - n_{\mathrm{v}} = (n_{\mathrm{c},1} + n_{\mathrm{c},2}) - (n_{\mathrm{v},1} + n_{\mathrm{v},2}) + 2. \tag{9}$$

From Eqs. (5)–(9), we have Eq. (4). □

Thus, the difference of the number of convex nodes and concave nodes is the same for flow patterns containing the same number of sinks, sources and saddle nodes. Given a couple of flow patterns containing the same number of sinks, sources and saddle nodes, we say one is simpler than the other if the number of convex and concave nodes of the former is smaller than that of the latter.

[8] The interested user might refer to [2, Proposition 1.8.4, p. 51], which describes the basic properties of the index of fixed points.

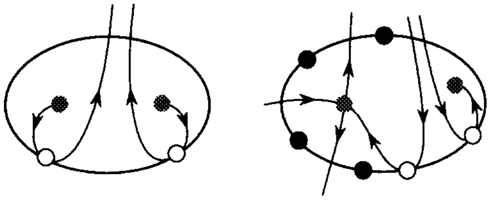


Fig. 12. Minimal flow patterns which cannot be derived from basic flow patterns without applying a distortion rule.

**Definition 3.** Let the flow indices of flow patterns $f$ and $g$ be $\langle n_v, n_c, n_{sa}, n_{ss} \rangle$ and $\langle n'_v, n'_c, n'_{sa}, n'_{ss} \rangle$, respectively. A flow pattern $f$ is said to be *simpler* than another flow pattern $g$ if $n_v < n'_v \wedge n_c < n'_c \wedge n_{sa} = n'_{sa} \wedge n_{ss} = n'_{ss}$.

Since $n_v$, $n_c$, $n_{sa}$, and $n_{ss}$ are computed independently, the above theorem can be used as a constraint for detecting numerical errors or missing information. It can also be used to enumerate the flow patterns efficiently. For example, if both $n_v$ and $n_c$ are 2, it follows from Eq. (4) that $n_{sa} = n_{ss} - 1$. Thus, we can start enumerating flow patterns from those with index $\langle 2, 2, 0, 1 \rangle$.

### *4.3. Enumerating flow patterns*

The flow grammar presented in the previous subsection is ambiguous in the sense that there usually exists more than one derivation which produces the same flow pattern. In order to cope with this unfortunate property, we have to rely on a generate and test method. In order to decrease the cost of enumeration, we use two techniques.

The first technique is to pose a constraint on the sequence of derivations so as to suppress a sequence of derivations which eventually produces a flow pattern to be generated otherwise. The second technique is at the representation level, as described in Section 5.2.

**Definition 4.** Flow pattern $P$ is *minimal* if there is no other flow pattern $Q$ such that $P$ results from applying a distortion rule to $Q$.

As for the twelve flow patterns shown in Fig. 11, only (1), (2), (4), (10), and (12) are minimal.

Let us consider the strategy of deriving all flow patterns with flow index $\langle n_v, n_c, n_{sa}, n_{ss} \rangle$. At a first glance, one might think that such flow patterns could be derived by firstly deriving minimal flow patterns with flow index $\langle -, -, n_{sa}, n_{ss} \rangle$ by repeatedly applying composition rules to basic flow patterns and then applying distortion rules until $n_v$ and $n_c$ become the given values. Unfortunately, such a naive procedure may fail to generate some flow patterns such as those shown in Fig. 12. The flow pattern shown in Fig. 12(a) cannot be derived by simply applying the composition rule to a couple of basic flow patterns; one has to apply a distortion rule first, as shown in Fig. 13.

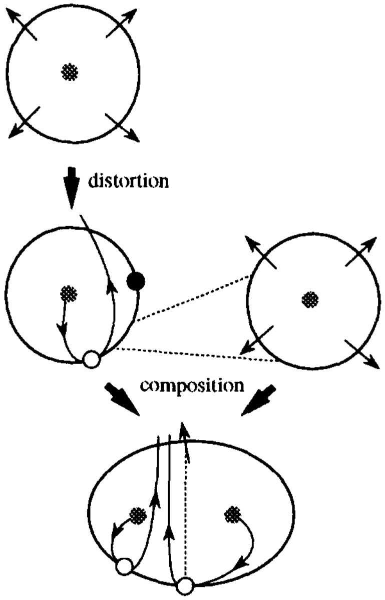


Fig. 13. Example of a derivation process for the flow pattern in Fig. 12(a).

However, we can know the upper bound of the number of applications of the distortion and the composition rules to derive any flow pattern with a given flow index, as described below.

**Proposition 5.** *Whenever a flow pattern $c$ is derived by applying the composition rule to flow patterns $a$ and $b$ and the internal segment of fusion $s$ of $a$ contains a couple of convex and concave nodes resulting from application of distortion rules, there exist other flow patterns $a'$ and $b'$ derivable by $FG1$ such that*

(i) *$a$ is derived from $a'$ by application of a distortion rule*;

(ii) *$b'$ is equivalent to $b$ except that $b'$ does not have a couple of points of contact on $s$*; *and*

(iii) *$c$ can be derived by applying the composition rule to $a'$ and $b'$.*

**Proof.** Without losing generality, we can assume that the flow pattern $c$ is derived from flow patterns $a$ and $b$ by applying the composition rule to the portion of interval $\overline{\alpha,\beta}$ of the segment of fusion $s$, as shown in Fig. 14(2). Consider partitioning $c$ along with a surface $s'$ which differs from $s$ only at the interval $\overline{\alpha',\beta'}$ where orbits are coherently cutting across the boundary (see Fig. 14(3)). Such partitioning is possible from the continuity of $C^1$ flow. Let the resulting flow patterns be $a'$ and $b'$ (see Fig. 14(1)). Flow pattern $a$ is derived by applying a distortion rule to $a'$. Flow pattern $b'$ is the same as $b$ except that there are no points of contact at the interval $\overline{\alpha',\beta'}$. It is generated by $FG1$, for it is also a structurally stable flow obtained by slightly perturbing the interval $\overline{\alpha,\beta}$ of $b$ and Proposition 1 holds. □

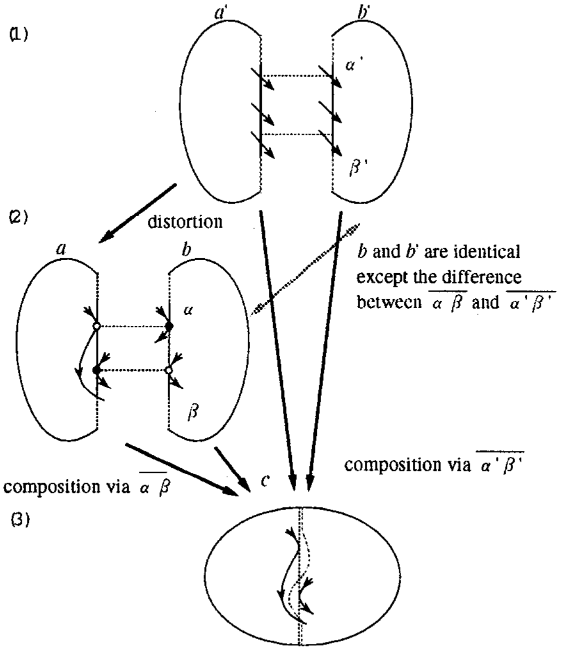

(1)
a'
b'
α'
β'
(2)
distortion
a
b
α
β
b and b' are identical except the difference between $\overline{\alpha\beta}$ and $\overline{\alpha'\beta'}$
composition via $\overline{\alpha\beta}$
c
composition via $\overline{\alpha'\beta'}$
(3)

Fig. 14. A case where flow pattern $c$ results from applying a distortion rule before applying the composition rule. In this case, the proposition suggests that there exists an alternative way of deriving $c$ from flow patterns $a'$ and $b'$ which are both simpler than $a$ and $b$, respectively.

The flow pattern $a$ in the above proposition is thought of as one resulting from applying a distortion rule before applying the composition rule. In this case, the proposition suggests that there exists an alternative way of deriving $c$ from flow patterns $a'$ and $b'$ which are both simpler than $a$ and $b$, respectively.

Similarly, there are several other cases in which the application of a distortion rule immediately prior to the application of the composition rule can be avoided by employing an alternative, simpler derivation process, as shown in Fig. 15. Unfortunately, there is a case as shown in Fig. 16 in which the composition occurs at the boundary edge resulting from the application of a distortion rule, and there are no ways of replacing the derivation process by a simpler one not containing the application of a distortion rule immediately prior to the application of the composition rule. Thus:

**Proposition 6.** *We cannot derive all flow patterns unless the application of a distortion rule is permitted prior to the application of the composition rule.*

However, it is also noted that peculiar cases in which the application of a distortion rule is necessary prior to the application of the composition rule are limited to the case shown in Fig. 16 and its variants. In such cases, the number of con-

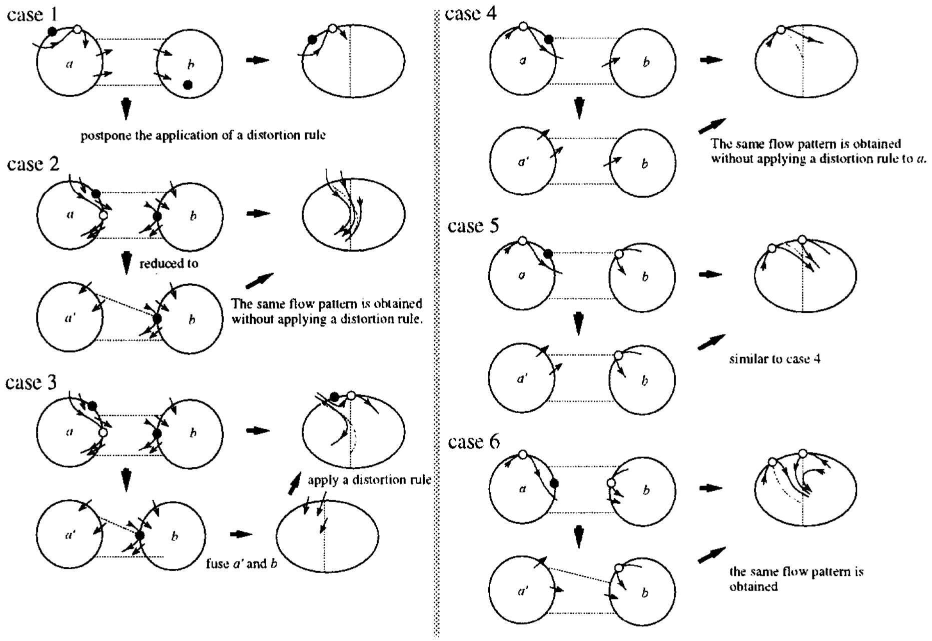

case 1
a
b
postpone the application of a distortion rule
case 2
a
b
reduced to
a'
b
The same flow pattern is obtained without applying a distortion rule.
case 3
a
b
apply a distortion rule
a'
b
fuse a' and b
case 4
a
b
a'
b
The same flow pattern is obtained without applying a distortion rule to a.
case 5
a
b
a'
b
similar to case 4
case 6
a
b
a'
b
the same flow pattern is obtained

Fig. 15. Cases in which the application of a distortion rule immediately prior to the application of the composition rule can be avoided by employing an alternative, simpler derivation process.

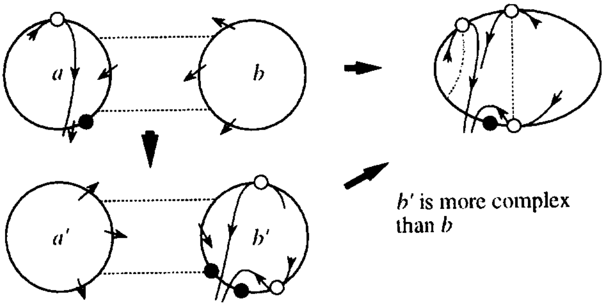


Fig. 16. The composition occurs at the boundary edge resulting from the application of a distortion rule, and there are no ways of replacing the derivation process by a simpler one not containing the application of a distortion rule immediately prior to the application of the composition rule.

cave nodes is at least one more than the total number of concave nodes of the two flow patterns to be fused. In addition, one concave node is introduced at the application of a distortion rule prior to the application of the composition rule. So we have:

**Proposition 7.** *A flow pattern with flow index $\langle n_v, n_c, n_{sa}, n_{ss} \rangle$ can be derived by applying a distortion rule at most $n_c$ times and the composition rule at most $(n_{sa} + n_{ss})$ times.*

### 4.4. Evaluation of a flow grammar

General requirements on a flow grammar are:

- *soundness*: any flow patterns generated by the flow grammar should correspond to a realizable flow;
- *completeness*: any flow patterns corresponding to an existing flow should be derived in finite steps; and
- *unambiguity*: any single flow pattern should not be derived in more than one way.

The flow grammar *FG*1 mentioned above is partly complete in the sense of Proposition 1, while it is not unambiguous as there might be the case in which a flow pattern may be derived in more than one way. This means that redundant computation is unavoidable to use *FG*1 as a knowledge source for predicting behavior. Although Proposition 7 suggests the upper bound of the number of steps for derivation, its effect is not powerful enough. We do not know so far how we might enumerate flow patterns in an efficient fashion. In addition, we need to devise an algorithm which can generate the set of all flow patterns consistent with observed geometric cues. Some kind of constraint satisfaction may help. We do not have any idea about the soundness of *FG*1, though our conjecture is positive.

## 5. Use of flow grammar in qualitative analysis of flow

We have implemented an algorithm for enumerating flow patterns and incorporated it into a system called PSX2NL.

### 5.1. Bottom-up analysis by PSX2NL

In normal situations, PSX2NL examines the flow in a bottom-up manner, attempting to build a qualitative description according to a prescribed fixed sequence: PSX2NL first examines whether the given flow has one or more fixed points; if so, it determines their type. If there is more than one fixed point in the given region, PSX2NL partitions the given region into several cells so that at most one fixed point may be contained in each cell; it then examines the geometric features of the flow, traces key orbits by numerical integration if necessary, generates a set of flow mappings for the given region, investigates the properties of the flow mappings, and derives conclusions about the qualitative behavior of the given region. Thus, an abstract description is gradually constructed from less abstract descriptions.

For example, Fig. 17 illustrates how PSX2NL analyzes the flow by Van der Pol's equation (2) in region *ABCD*, in normal situations. PSX2NL derives a set of flow mappings (3) for the cell *ABEF* and

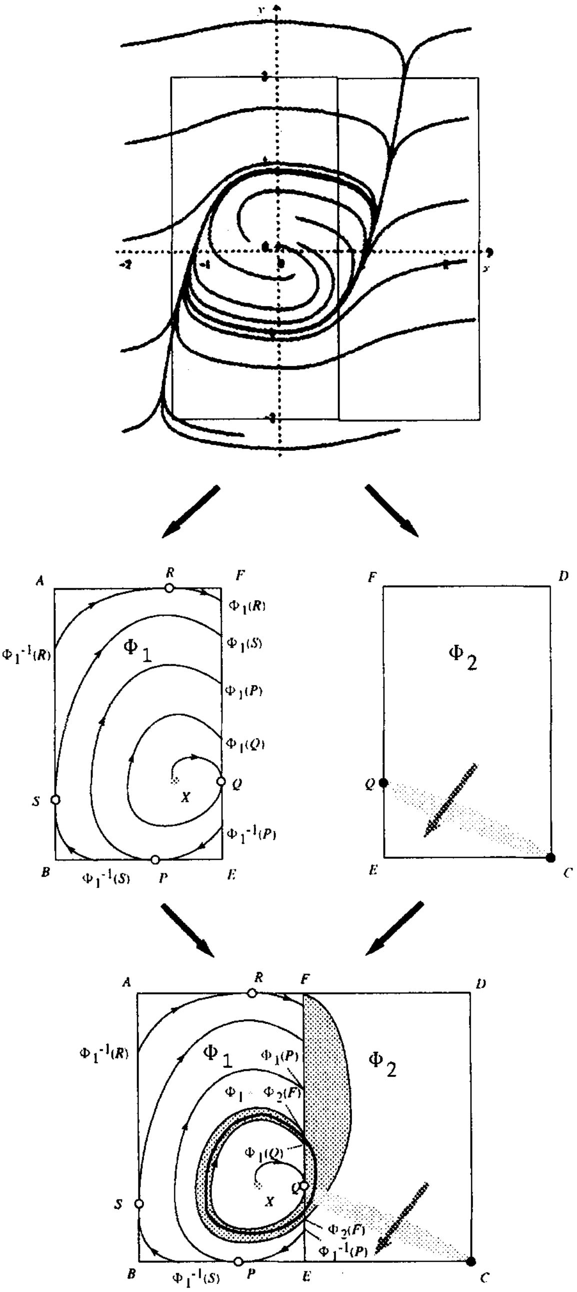


Fig. 17. Reasoning about qualitative behavior with flow mappings.

$$\Phi_2 : \overline{C,D,F,Q} \to \overline{Q,E,C} \tag{10}$$

for the cell *FECD*.

By combining the two sets of flow mappings for the two cells, PSX2NL obtains

$$\Phi_2(\overline{F,Q}) = \overline{\Phi_2(F),Q} \subset \overline{\Phi_1^{-1}(P),Q}, \tag{11}$$

$$\Phi_1(\overline{\Phi_1^{-1}(P),Q}) = \overline{\Phi_1(Q),\Phi_1(P)} \subset \overline{F,Q}, \tag{12}$$

and hence,

$$\Phi_1 \circ \Phi_2(\overline{F,Q}) \subset \overline{F,Q}. \tag{13}$$

Since Eq. (13) suggests that all orbits passing through the interval $\overline{F,Q}$ never leave the interval, PSX2NL concludes from the Poincaré-Bendixon theorem[9] that there exists an attracting bundle $\Phi_C$ of orbits containing at least one limit cycle that is transverse to $\Phi_1 \circ \Phi_2(\overline{F,Q})$.[10] It also follows that all orbits passing through the interval $\overline{F,Q}$ tend towards $\Phi_C$ as $t \to \infty$.

### 5.2. *Top-down analysis by PSX2NL*

If something goes wrong and the standard sequence turns out to be intractable, the analysis process switches to the top-down mode, trying to find the most plausible interpretation that matches the observations made so far. As a knowledge source, PSX2NL uses the flow grammar *FG*1. The flow grammar provides PSX2NL with theoretical constraints, allowing it to operate in a top-down manner.

The top-down analysis is basically a generate and test method. The input is a set of *observations* consisting of partially traced orbits and estimated location of convex and concave nodes. A flow pattern $P$ is an *interpretation* of an *observation* $O$ if there exists (possibly empty) a set of *assumptions* $A$ such that $O = P \cup A$. An interpretation is *minimal* when there is no other interpretation which explains the observation with a smaller set of assumptions. PSX2NL uses the enumerator described in the previous section to generate flow patterns in turn and seeks for a minimal interpretation.

For example, given the observation shown in Fig. 18, PSX2NL produces twelve minimal interpretations, two of which are shown in Fig. 19. Currently, PSX2NL will simply increase the number of observations when more than one minimal interpretation is found. Whenever PSX2NL detects a symptom suggesting that a limit cycle is contained in the given region, it will divide the region into two by a line across the limit cycle like orbit. Focusing observation for resolving ambiguity would be an interesting problem left for future.

PSX2NL has additional features as described below.

First, the flow grammar allows PSX2NL to predict the existence of key orbits and to plan numerical computation to find their location. For example, given an observation

[9] See [3, p. 248] for more details.

[10] Note that $\Phi_1 \circ \Phi_2(\overline{F,Q}) = \overline{\Phi_1(Q),\Phi_1 \circ \Phi_2(F)}$.

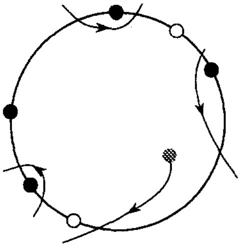


Fig. 18. Observation (drawn schematically).

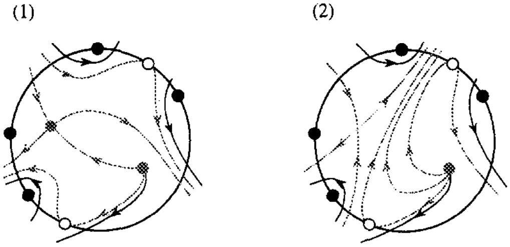


Fig. 19. Minimal interpretations of the observation in Fig. 18.

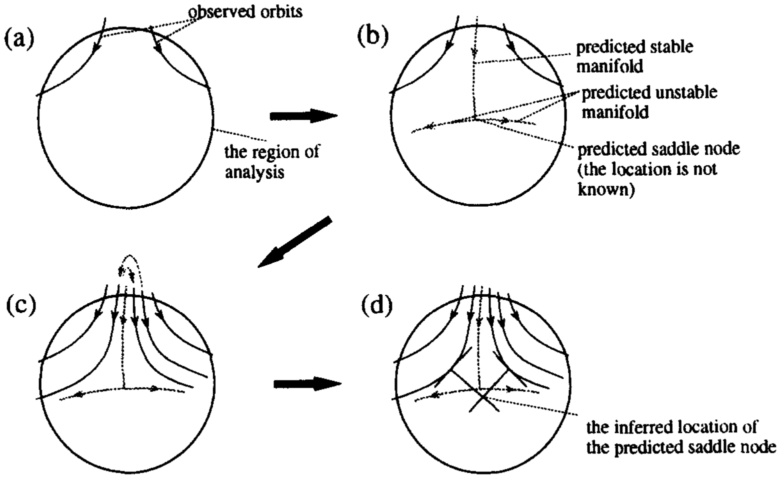
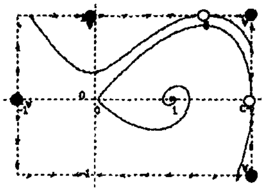

(a)
observed orbits
the region of analysis
(b)
predicted stable manifold
predicted unstable manifold
predicted saddle node (the location is not known)
(c)
(d)
the inferred location of the predicted saddle node

Fig. 20. Planning numerical computation.

as illustrated in Fig. 20(a), PSX2NL predicts the existence of a saddle node and attempts to find the location in the phase space (Fig. 20(b)), infers the location at which invariant manifolds of the predicted saddle node intersect the boundary of a given region by narrowing down the envelop enclosing the saddle node and associated invariant manifolds (Fig. 20(c)), and finally it infers the location of the saddle node (Fig. 20(d)).

(a) Observation

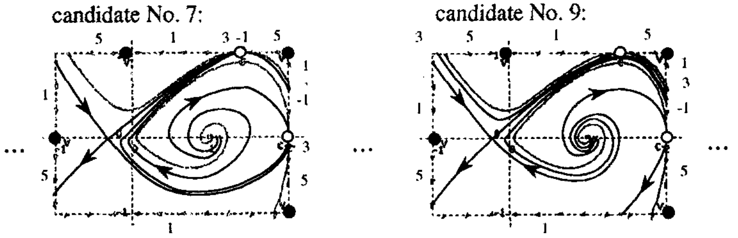


search for interpretation

```
f-rep inferred: ((- 1) (+ 5 1) (- 5 1) (+ 5) (- -1 1) (+ 5))
```

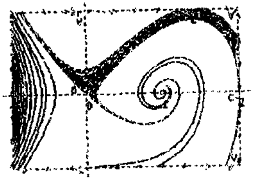
(b) Searching for flow patterns which explain the observation; candidates are listed from the simplest ones; in this case, twelve equally simple candidates are found;

```
12 patterns match.
> their f-reps are:
No. 1: ((- -1 1) (+ 5) (- -1 1) (+ 5 1) (- 5 1) (+ 5))
No. 2: ((- -1 1) (+ 5 3) (- 1) (+ 5 1) (- 5 3 1) (+ 5))
No. 3: ((- -1 3 1) (+ 5) (- 1) (+ 5 3 1) (- 5 1) (+ 5))
No. 4: ((- 5 3 1) (+ 5) (- 1 -1 1) (+ 5 5 3) (- 1) (+ 5 1))
No. 5: ((- 5 1) (+ 5) (- 3 1 -1 1) (+ 5 5) (- 1) (+ 5 3 1))
No. 6: ((- 5 -1 1) (+ 5) (- -1 1) (+ 5) (- -1 1) (+ 5 1))
No. 7: ((- -1 1) (+ 5) (- -1 3 1) (+ 5 1) (- 5 1) (+ 5 3))
No. 8: ((- 5 3 -1 1) (+ 5) (- -1 1) (+ 5 3) (- 1) (+ 5 1))
No. 9: ((- 5 -1 1) (+ 5) (- -1 3 1) (+ 5) (- 1) (+ 5 3 1))
No. 10: ((- 5 1) (+ 5) (- 3 -1 1) (+ 5) (- -1 1) (+ 5 3 1))
No. 11: ((- -1 5 1) (+ 5) (- 3 1) (+ 5 1) (- 5 1) (+ 5 3 1))
No. 12: ((- 5 1) (+ 5) (- 3 -1 3 1) (+ 5) (- 1) (+ 5 3 3 1))
```


candidate No. 7:

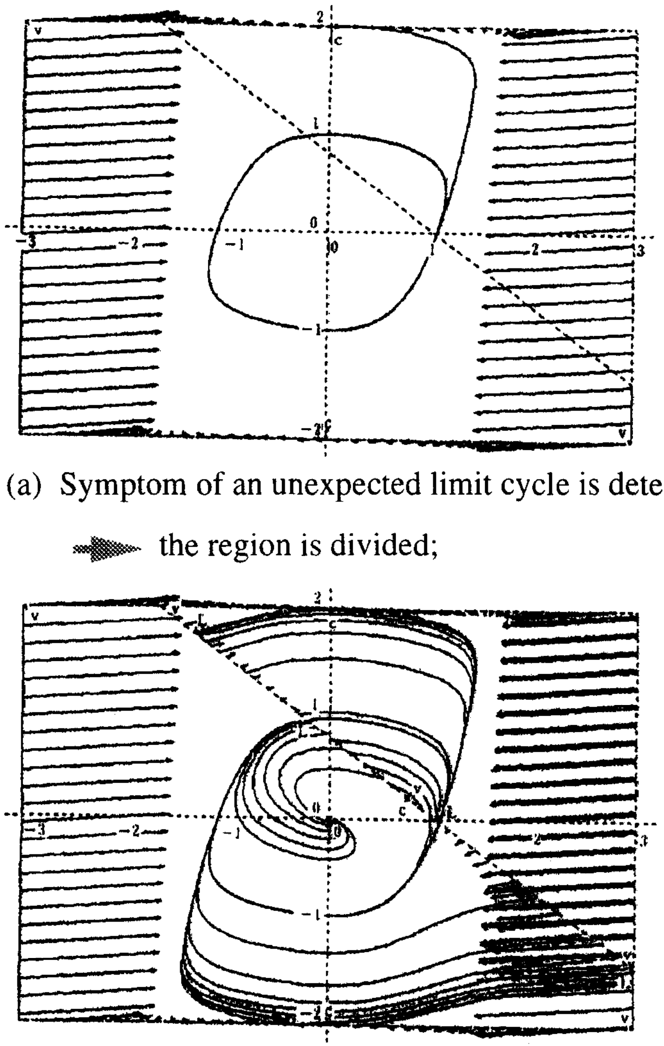
candidate No. 9:

(c) In order to resolve ambiguity, numerical computation is planned and executed;



(d) Candidate No. 9 is selected as the most plausible interpretation.

Fig. 21. Focusing numerical computation.



(a) Symptom of an unexpected limit cycle is detected

➡ the region is divided;

(b) Flow in each region is analyzed individually and the results are merged.

Fig. 22. Rearranging the analysis process when an unexpected result is obtained.

Second, the flow grammar permits PSX2NL to focus numerical computation. For example, if there is more than one possible interpretation of an observation (upper half of Fig. 21), PSX2NL will plan numerical computation that is expected to resolve the ambiguity (lower half of Fig. 21).

Third, the flow grammar enables PSX2NL to rearrange the analysis process when an unexpected result is obtained. For example, when a symptom of an unexpected

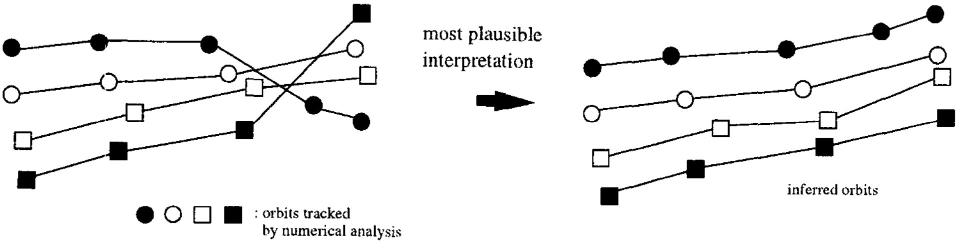


Fig. 23. Correcting inconsistency due to numerical errors.

limit cycle is observed, as in Fig. 22(a), PSX2NL will divide a region into cells and try to prove the existence of a limit cycle by numerical and symbolic computation (Fig. 22(b)).

Finally, the flow grammar helps to detect inconsistent numerical results and propose a plausible interpretation. For example, nonintersection constraints of orbits may be violated because of numerical errors, as in Fig. 23(a). In such cases, PSX2NL suggests the most plausible interpretation (Fig. 23(b)).

To reduce the cost of comparing flow patterns, we use a short-hand representation of flow patterns. An *f-rep* is a cyclic list[11] of the form $\langle\langle \ldots, s_i[x_{i,1}, \ldots, x_{1,m_i}], \ldots \rangle\rangle$. Each element of an f-rep corresponds to a boundary segment delimited by a couple of points of contact and qualitatively denotes in the counterclockwise order how points on the boundary segment are mapped by orbit intervals involved in the flow pattern. If the interval corresponding to $x_{i,j}$ is mapped from/to another boundary segment $B$, we use a positive integer indicating the relative position of $B$ counted in the counterclockwise order from the current boundary segment. If it is either a fixed point or a limit cycle, we use a negative integer. $s_i$ denotes the orientation of the flow there: it is "+" if the flow comes from the outside, and "−" if it leaves for the outside.

For example, an f-rep for the flow pattern in Fig. 24 is:

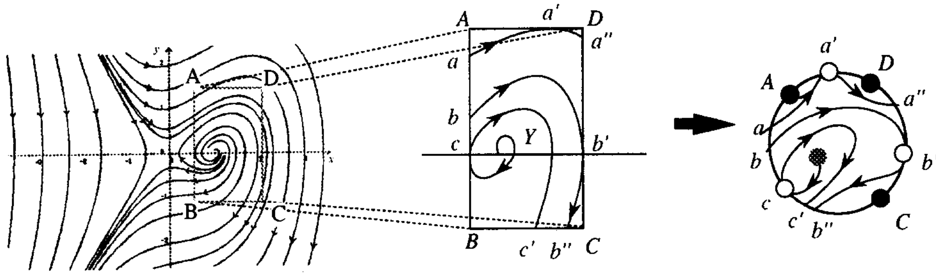
$$\langle\langle +[5,3,1], -[-1,5,1], +[5], -[3,1], +[5], -[1] \rangle\rangle, \tag{14}$$

where the first element is for boundary segment $\overline{Ac}$ and the second is for $\overline{cC}$, and so on. Note that f-rep for a set of flow mappings is uniquely defined *except* the existence of variants which only differ from each other in the way fixed points are numbered. And importantly it seems that different flow mappings give different f-reps. More study is left for future.

### 5.3. Implementation issues

For the procedure illustrated in the previous subsection to work, numerical and symbolic computation and a high-level qualitative reasoning procedure should closely interact with each other. It should be noted that neither low-level procedures nor high-level

[11] We denote a cyclic list consisting of $x_1, \ldots, x_n$ as $\langle\langle x_1, \ldots, x_n \rangle\rangle$. By definition, $\langle\langle x_1, \ldots, x_n \rangle\rangle = \langle\langle x_2, \ldots, x_n, x_1 \rangle\rangle = \cdots = \langle\langle x_n, x_1, \ldots, x_{n-1} \rangle\rangle$.



where the flow is governed by:

$$\begin{cases} \frac{\mathrm{d}x}{\mathrm{d}t} = y \\ \frac{\mathrm{d}y}{\mathrm{d}t} = x - x^2 + 0.2y + 0.3xy \end{cases}$$

Fig. 24. A nonlinear flow and a flow pattern representing its portion.

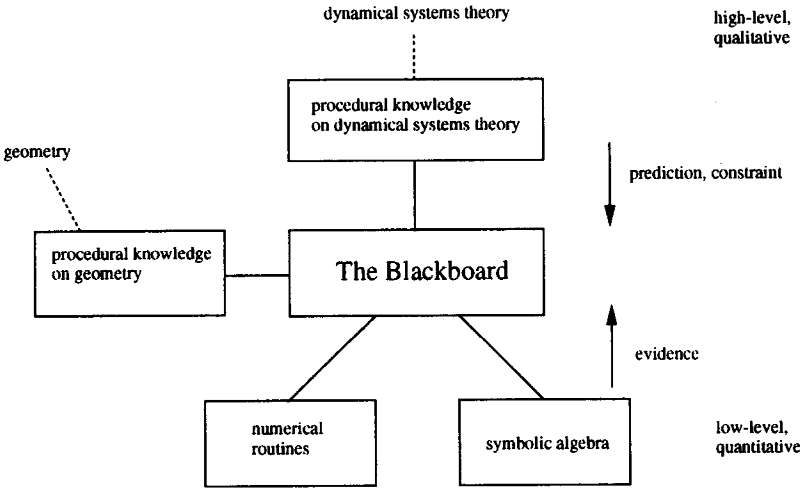


Fig. 25. A blackboard model as a basis of the control scheme of PSX2NL.

procedures are complete; high-level procedures are incomplete in the sense that they cannot yield any conclusion unless evidence is provided, whereas low-level procedures are incomplete in the sense that they cannot determine how accurate an answer is they need to produce in order to avoid missing some important solution. In order to facilitate intensive coupling of the processes at different levels of abstraction, we have employed a blackboard model as a basis of a control scheme (Fig. 25). Thus, the system has a shared memory space (the blackboard) and a library of processes (KSs: knowledge sources). KSs interact indirectly by updating the contents of the blackboard. The blackboard model makes it possible to implement multiple strategies.

To allow the complex control structure to be implemented easily, we have employed the following features:

(i) explicit representation of the control structure on the blackboard so that KSs can directly access and manipulate it;

(ii) uniform representation of all objects on the blackboard as attribute-value pairs.

On the other hand, our approach to designing and implementing PSX2NL has the following limitations:

(i) all knowledge used for problem solving must be represented in a procedural form as KSs;

(ii) the terminology for representing objects and relations has not been carefully designed;

(iii) no attempt has been made to incorporate intuitive guidance, possibly obtained from statistics.

Thus, we have not addressed the horizontal coordination problem in designing and implementing PSX2NL.

## 6. Related work

Attempts to incorporate global information have been made by several authors [4, 10]. Unfortunately, most computational methods developed so far only make use of the nonintersection constraint of orbits and some other partial constraints, and hence the ability of reasoning about global behaviors is still quite limited. Their weakness mostly comes from the lack of adequate representation. In contrast, flow patterns and a flow grammar provide a means for reasoning about various aspects of geometric constraints, allowing an envisioner to symbolically reason about the structure of global and asymptotic behaviors.

Intelligent analysis of nonlinear ODEs is quite a new field. POINCARE [9] would be the first program addressing intelligent analysis of nonlinear ODEs. POINCARE integrates qualitative and quantitative methods, as PSX2NL does. Both POINCARE and PSX2NL work on planar ODEs including nonlinear ODEs, though POINCARE supports bifurcation analysis which is not yet implemented in PSX2NL. The difference in phase portrait analysis is that PSX2NL makes use of more representation than POINCARE and other programs based on the conventional simulation technology. This leads to three consequences. First, PSX2NL saves computational resources, for it keeps geometric and topological information in an more abstract form. For example, PSX2NL keeps information about only a few essential points on orbits, while POINCARE has to keep the location of all points on orbits. Second, PSX2NL can derive richer conclusion from the same observation obtained by quantitative analysis, as demonstrated in the previous section. Third, PSX2NL is more robust from incompleteness of information and numerical errors. For example, POINCARE relies on an external package in locating fixed points. If the package fails POINCARE fails, too. In contrast, PSX2NL can switch to a robust, approximate method based on a flow grammar.

A stochastic approach [1] is another candidate of uniform treatment of global behavior supported by mathematical theories. Since the stochastic approach possesses a complementary nature to ours, it would be interesting to seek a way for combining the two.

## 7. Conclusion

I have presented a *flow grammar*, a grammatical specification of all possible patterns of solution curves one may see in the phase space. I have described *flow pattern*, a semi-symbolic representation of the patterns of solution patterns in the phase space and studied the properties of the flow grammar. Finally, I have described how the flow grammar is used in qualitative analysis to plan, monitor, and interpret the result of numerical computation.

An important work left for future research is the extension into higher dimensional flow. The research in that direction is quite challenging both theoretically and practically. Unfortunately, the extension of this work into higher dimensional flows is not trivial, for firstly flow patterns become far more complicated, secondly, representing higher dimensional geometric objects is hard, and thirdly it becomes subtle to characterize the topological structure of flow. However, we believe that the concepts exploited in this paper would be of much help in such extension.

## References

[1] J. Doyle and E.P. Sacks, Stochastic analysis of qualitative dynamics, in: *Proceedings IJCAI-89*, Detroit, MI (1989) 1187–1192.

[2] J. Guckenheimer and P. Holmes, *Nonlinear Oscillations, Dynamical Systems, and Bifurcations of Vector Fields* (Springer, Berlin, 1983).

[3] M.W. Hirsch and S. Smale, *Differential Equations, Dynamical Systems, and Linear Algebra* (Academic Press, New York, 1974).

[4] W.W. Lee and B.J. Kuipers, Non-intersection of trajectories in qualitative phase space: a global constraint for qualitative simulation, in: *Proceedings AAAI-88*, St. Paul, MN (1988) 286–290.

[5] M. Leyton, A process-grammar for shape, *Artif. Intell.* **34** (1988) 213–247.

[6] T. Nishida and S. Doshita, A geometric approach to total envisioning, in: *Proceedings IJCAI-91*, Sydney (1991) 1150–1155.

[7] T. Nishida, K. Mizutani, A. Kubota and S. Doshita, Automated phase portrait analysis by integrating qualitative and quantitative analysis, in: *Proceedings AAAI-91*, Anaheim, CA (1991) 811–816.

[8] T. Nishida and S. Doshita, Qualitative analysis of behavior of systems of piecewise linear differential equations with two state variables, *Artif. Intell.* **75** (1995) 3–29.

[9] E.P. Sacks, Automatic analysis of one-parameter planar ordinary differential equations by intelligent numeric simulation, *Artif. Intell.* **48** (1991) 27–56.

[10] P. Struss, Global filters for qualitative behaviors, in: *Proceedings AAAI-88*, St. Paul, MN (1988) 275–279.

[11] K.M.-K. Yip, Generating global behaviors using deep knowledge of local dynamics, in: *Proceedings AAAI-88*, St. Paul, MN (1988) 280–285.