# Path Problems in Structured Graphs

M. ANCONA*, L. DE FLORIANI* AND J. S. DEOGUN†

* *Istituto per la Matematica Applicata del C.N.R., Via L. B. Alberti 4, 16132 Genova, Italy*

† *Department of Computer Science, University of Nebraska, Lincoln, NE 68588, U.S.A.*

***A structured graph is a hierarchy of graphs which provides a representation of a graph at variable detail levels. In this paper we investigate the path problem in a structured graph. The concept of structured graph is defined, and a formal definition of path in such a structure is given. The relationship between structured paths and canonical ones in the graph represented by a structured graph is investigated. Algorithms to construct structured paths and to compute a structured path from a given canonical one are presented.***

*Received December 1984*

## 1. INTRODUCTION

Hierarchical graph models have been used as a tool for the design and analysis of systems in a variety of disciplines. In Ref. 4 the authors study a hierarchical graph structure, called structured graph,[1] at an abstract level, and develop a formal notation for such a structure and its properties. A structured graph is defined, intuitively, as a graph model, in which arcs or vertices can in turn represent structured graphs, thus leading to a hierarchy of graphs, in which higher-level graphs correspond to a greater degree of detail in the graph model specification.

Structured graphs find application in VLSI system design,[8,10] computer network description and analysis,[5,13,15] Petri nets,[14] flow graphs and software system models,[6,16] project network analysis,[2] and the development of parallel algorithms for graph manipulation.[12] The main motivation to the use of structured graphs lies in the possibility of a modular and structured approach to the development of large graph-based models as demonstrated by the use of a system for large-size graph manipulation based on structured graphs developed by two of the authors.[3] Essentially, structured graphs combine the two concepts of substitution decomposition of a digraph[9,11] and of modular representation at different levels of abstraction.

A structured graph is defined as a pair $\gamma = (\mathcal{G}, T)$, where $\mathcal{G} = \{G_i \mid i = 1, 2, \ldots, n\}$ is a finite family of graphs, called components of $\gamma$, and $T$ is a rooted tree describing the parent–child relationship among the elements of $\mathcal{G}$. The notion of structured graph naturally leads to the concept of structured solution. A structured solution to a given graph problem in the graph represented by a structured graph is a solution which conforms to the hierarchical organisation of the structured graph, in such a way that a confinement of this solution to any component in the hierarchical representation is a well-defined solution for that component. In this paper we study the properties of paths in structured graphs. In particular, we define the concepts of path in a structured graph $\gamma$ and of structured path in the graph represented by $\gamma$, called completely expanded graph associated with $\gamma$. Structured paths are the basic tool for studying path problems in models based on structured graphs. For instance, when applying a structured approach to the development of VLSI systems, structured paths correctly describe interconnection paths among system components, among subcomponents within every component, and so on. Similarly, in computer network analysis, structured paths describe main communication paths among clusters, and inside each cluster among subclusters or lower-level elements. These applications motivate the study of the properties of structured paths and of the relationship between structured and canonical paths in a graph admitting a hierarchical representation in a structured graph form.

The remainder of this paper is organised as follows. In Section 2 the definition of structured graph is introduced, and several concepts and properties related to structured graphs are developed. In Section 3 the definition of path in a structured graph is given, and Section 4 presents properties and algorithms related to paths. Section 5 summarizes the results.

## 2. STRUCTURED GRAPHS: DEFINITIONS AND PROPERTIES

Most of the graph terminology used in this paper is standard (see e.g. Ref. 7). A graph is defined as a pair $G = (V, E)$, where $V$ is the set of vertices and $E$ is the set of pairs of distinct vertices called arcs. If the arcs are ordered pairs of vertices, the graph is said to be directed, otherwise it is called undirected. In the following we introduce the concepts of vertex- and arc-reducibility of a subgraph for both directed and undirected graphs, which are the basis of the notion of structured graph (see also Ref. 4).

Let $G = (V, E)$ be a graph and $H = (V_H, E_H)$ a connected subgraph of $H$. The set of boundary vertices of $H$, $B(H)$, is defined as follows:

$$B(H) = \{y \in V_H \mid \text{there exists } x \in V - V_H \text{ such that } (x, y) \in E\}$$

If $G$ is a digraph, for every connected subgraph $H$ of $G$, the input vertex set $I(H)$ and the output vertex set $O(H)$ are defined as follows:

$$I(H) = \{y \in V_H \mid \text{there exists } x \in V - V_H \text{ such that } (x, y) \in E\}$$

$$O(H) = \{x \in V_H \mid \text{there exists } y \in V - V_H \text{ such that } (x, y) \in E\}$$

A path (or cycle) $P$ in $G$ is said to be internal to $H$ if and only if every arc of $P$ belongs to $H$. $H$ is said to be traversable if and only if there exists an internal path in $H$ from $x$ to $y$ for every $x, y \in B(H)$. Similarly, if $G$ is a digraph and $H$ a connected subgraph of $G$, $H$ is said to be traversable if and only if there exists an internal path in $H$ from $x$ to $y$ for every $x, y \in V_H$, $x \in I(H)$ and $y \in O(H)$.

Let $G = (V, E)$ be a graph. A subgraph $G_H = (V_H, E_H)$ of $G$ is said to be reducible to a vertex if and only if the following two properties are satisfied:

(i) $G_H$ is traversable

and

(ii) $(x_1, y), (x_2, y) \in E$, $x_1, x_2 \in V_H$, $y \notin V_H$ implies that $x_1 = x_2$. Let $\{G_i \mid i = 1, 2, \ldots, n\}$, $G_i = (V_i, E_i)$ be a family of disjoint subgraphs of $G$. Subgraphs $G_i, i = 1, 2, \ldots, n$ are said to be mutually reducible to single vertices, respectively, if and only if each $G_i$ is reducible to a vertex and the following condition is satisfied:

there exists at most one arc $(x, y)$ [or $(y, x)$] $\in E$ such that $x \in \bar{V} [y \in \bar{V}]$ and

$$y \in V_k [x \in V_k],\ k = 1, 2, \ldots, n,$$

where

$$\bar{V} = V - \bigcup_{i=1}^{n} V_i.$$

The graph obtained from $G$ by reducing each subgraph $G_i$, $i = 1, 2, \ldots, n$ to a vertex $v_i$ is called the reduced graph associated with $G$ and defined by the family of subgraphs $\{G_i \mid i = 1, 2, \ldots, n$. The reduced graph is denoted by $G^{(R)} = (V^{(R)}, E^{(R)})$, where

$$V^{(R)} = (V - \bigcup_{i=1}^{n} V_i) \cup \{V_i \mid i = 1, 2, \ldots, n\}$$

and

$$\begin{aligned} E^{(R)} = E \cap (V^{(R)} \times V^{(R)}) \cup \{&(v_i, v_j) \mid \text{there exists } (x, y) \in E, \\ &x \in B(V_i) \text{ and } y \in B(V_j)\} \cup \\ \{&(x, v_k) \mid x \in \bar{V} \text{ and there exists } (x, y) \in E, \\ &y \in B(V_k)\} \cup \\ \{&(v_k, y) \mid y \in \bar{V} \text{ and there exists } (x, y) \in E, \\ &x \in B(V_k)\} \end{aligned}$$

The reduced graph $G^{(R)}$ associated with a directed graph $G$ is defined similarly by replacing $x \in B(V_i)$ with $x \in O(V_i)$ and $y \in B(V_i)$ with $y \in I(V_i)$. The vertex $v_i$ in $V^{(R)}$ obtained by reduction of subgraph $G_i$ is called a macrovertex. Conversely, every vertex in $V - \bar{V}$ is called a simple vertex of $G^{(R)}$.

The purpose of the traversability property in vertex-reducible graphs is to ensure that every subgraph $G_i$ can be correctly represented as a vertex in the reduced graph $G^{(R)}$, whereas properties (ii) and (iii) make sure that $G^{(R)}$ does not have multiple arcs. This assumption is made only in order to simplify the treatment of structured graphs as well as the description of the algorithms working on them. The extension to multigraphs is, however, straightforward.

Let $H = (V_H, E_H)$ be a graph and $\mathcal{G} = \{G_i \mid G_i = (V_i, E_i), i = 1, 2, \ldots, n\}$ be a family of disjoint graphs. Define $M_v \subset V_H$ such that each $G_i$ in $\mathcal{G}$ is associated with a unique $h_i \in M_v$ by a mapping $\mu$. Define two mappings $\phi_i$ and $\psi_i$ over the set of arcs of $H$ incident to and from $h_i$ as follows:

$$\phi_i((x, h_i)) = v \quad \text{where} \quad x \in V_H - \{h_i\} \quad \text{and} \quad v \in V_i,$$

$$\psi_i((h_i, y)) = w \quad \text{where} \quad y \in V_H - \{h_i\} \quad \text{and} \quad w \in V_i.$$

Again, define the sets of the input and output vertices of $G_i$ in $G$, denoted by $I(G_i)$ and $O(G_i)$ respectively, as follows:

$$I(G_i) = \{v \mid v \in V_i \text{ and } \phi_i((x, h_i)) = v, x \in V_H\}$$
$$O(G_i) = \{w \mid w \in V_i \text{ and } \psi_i((h_i, y)) = w, y \in V_H\},$$
$$i = 1, 2, \ldots, n.$$

If every $G_i$ in $\mathcal{G}$ is traversable (with respect to $I(G_i)$ and $O(G_i)$), then the graph $G = (V, E)$ such that

$$V = \bigcup_{i=1}^{n} V_i \cup (V_H - M_v)$$

$$\begin{aligned} E = \bigcup_{i=1}^{n} E_i \cup (E_H \cap M_v \times M_v) \cup \{(x, y) \mid &\text{ either} \\ &y = \phi_i((x, h_i)), \\ &(x, h_i) \in E_H \text{ or } y = \psi_i((h_i, y)), \ (h_i, y) \in E_H\} \end{aligned}$$

is called the expanded graph of $H$ defined by family $\mathcal{G}$, mapping $\mu$ and the two families of mappings $\Phi = \{\phi_i\}_{i \leqslant n}$ and $\Psi = \{\psi_i\}_{i \leqslant n}$. Every $G_i$ in $\mathcal{G}$ is called the expansion graph of the vertex $h_i$ in $M_v$.

If $H$ is an undirected graph, the definition of an expanded graph is slightly different. Since the arcs have no orientation, $\phi_i((x, h_i)) = \psi_i((h_i, x))$, $x \in V_H$ and $h_i \in M_v$. Hence only one mapping $\phi_i = \psi_i$ needs to be defined, and $B(G_i) = I(G_i) = O(G_i)$ is thus the set of boundary vertices of graph $G_i$ in $\mathcal{G}$.

An immediate consequence of the definitions of reduced and expanded graph is that if $G^{(R)}$ is the reduced graph associated with $G$ and defined by the family $\mathcal{G}$, $G$ is the expanded graph of $G^{(R)}$ defined by family $\mathcal{G}$ by mapping $\mu$ and by the families of functions $\phi$ and $\psi$, where

(i) for every $G_i$ in $\mathcal{G}$ there exists a macrovertex $h_i \in V^{(R)}$ such that $\mu(G_i) = h_i$;

(ii) for every macrovertex $h_i$ in $V^{(R)}$ $\phi_i((x, h_i)) = v$, where

$$(x, h_i) \in E^{(R)}, \quad v \in I(G_i) \quad \text{and} \quad (x, v) \in E$$

and $\psi_i((h_i, y)) = w$, where

$$(h_i, y) \in E^{(R)}, \ w \in O(G_i) \quad \text{and} \quad (w, y) \in E.$$

Hence two operators $V_{\text{Exp}}$ and $V_{\text{Red}}$, called respectively vertex expansion and vertex reduction can be defined. The expansion operator maps the reduced graph $G^{(R)}$ and the family $\mathcal{G} = \{G_i \mid i = 1, 2, \ldots, n\}$ into the expanded graph $G$. More formally

$$V_{\text{Exp}}(G^{(R)}, \mathcal{G}, \mu, \phi, \psi) = G.$$

Conversely, the reduction operation maps a graph $G$ and a family $= \{G_i \mid i = 1, 2, \ldots, n\}$ of mutually reducible subgraphs of $G$ into the reduced graph $G^{(R)}$, that is

$$V_{\text{Red}}(G, \mathcal{G}) = (G^{(R)}, \mu, \phi, \psi),$$

where $\mu$, $\phi$ and $\psi$ are defined by $G$ and $\mathcal{G}$ as above.

Let $G = (V, E)$ be a digraph; a subgraph $G_i = (V_i, E_i)$ is said to be reducible to an arc if and only if

(i) $I(G_i) = \{x_i\}$, $O(G_i) = \{y_i\}$, $x_i, y_i \in V_i$, $x_i \neq y_i$;

(ii) every vertex $z$ in $V_i - \{x_i, y_i\}$ belongs to any internal path;

(iii) $(x_i, y_i)$ is not in $E - E_i$;

(iv) $x_i, y_i$ do not belong to any internal cycle in $G_i$.

Let $\{G_i \mid i = 1, 2, \ldots, n\}$, $G_i = (V_i, E_i)$, be a family of connected subgraphs of $G$. Then subgraphs $G_i$, $i = 1, 2, \ldots, n$ are said to be mutually reducible to single arcs if and only if each $G_i$ is arc-reducible and the following two conditions are satisfied:

(v) $E_i \cap E_j = \varnothing$, for $i \neq j$, $i = 1, 2, \ldots, n$;
(vi) $I(G_i) \times O(G_i) \neq I(G_j) \times O(G_j)$, for $i \neq j$, $i = 1, 2, \ldots, n$.

Vertices $x_i$ and $y_i$ in $V_i$ are respectively called the input and output vertex of graph $G_i$. The set $B(G_i) = \{x_i, y_i\}$, is called the set of boundary vertices of $G_i$. The two latter conditions in the above definition ensure that two graphs $G_i$ and $G_j$ in $\mathscr{G}$, $i \neq j$, have no edge and no internal vertex in common, i.e.: $V_i - \{x_i, y_i\} \cap V_j - \{x_j, y_j\} = \varnothing$, whereas it can happen that the input vertex of $G_i$ is the same as the output vertex of $G_j$ or conversely, but never $x_i = x_j$ and $y_i = y_j$.

If $G$ is an undirected graph, the conditions in the definitions of arc reducibility and mutual reducibility above must be replaced with the following ones:

(i) for every $G_i$ in $\mathscr{G}$, $B(G_i) = \{x_i, y_i\}$, $x_i, y_i \in V_i$, $x_i \neq y_i$;
(ii) every vertex $z$ in $V_i - B(G_i)$ belongs to an internal path in $G_i$ joining the two boundary vertices;
(iii) $(x_i, y_i)$ is not in $E - E_i$;
(iv) $E_i \cap E_j = \varnothing$ for $i \neq j$;
(v) $B(G_i) \cap B(G_j) = B$ where $B = \varnothing$ or $B = \{v\}$

Thus the graph $G^{(R)} = (V^{(R)}, E^{(R)})$ such that

$$V^{(R)} = \bigcup_{i=1}^{n} B(G_i) \cup (V - \bigcup_{i=1}^{n} V_i)$$

$$E^{(R)} = (E - \bigcup_{i=1}^{n} E_i) \cup E^{(M)}$$

where

$$E^{(M)} = \{z_i, z_i = (x_i, y_i), i = 1, 2, \ldots, n\}$$

is called the reduced graph associated with $G$ and defined by family $\mathscr{G} = \{G_i \mid i = 1, 2, \ldots, n\}$. The traversability property in both definitions of arc reducibility, together with property (iv) for digraphs, ensures that every subgraph $G_i$ in $\mathscr{G}$ can be correctly represented by an arc $(x_i, y_i)$ in the reduced graph $G^{(R)}$, and properties (iii) and (v) ensure that $G^{(R)}$ contains no multiple arc. Thus every arc $a$ in $E^{(R)}$ such that $a \in E^{(M)}$ is called a macroarc. Conversely, every arc in $E^{(R)} - E^{(M)}$ is called a simple arc of $G^{(R)}$.

Let $H = (V_H, E_H)$ be a graph and $\mathscr{G} = \{G_i \mid G_i = (V_i, E_i), i = 1, 2, \ldots, n\}$ a family of graphs such that

(i) there exists a mapping $\mu: \mathscr{G} \rightarrow E_H$, such that $\mu(G_i) = \{(x_i, y_i)\}$;
(ii) $E_i \cap (E_H - \{x_i, y_i\}) = \varnothing$ and $V_i \cap V_H = \varnothing$, $i = 1, 2, \ldots, n$;
(iii) $E_i \cap E_j = \varnothing$ and $(V_i - \{x_i, y_i\}) \cap (V_j - \{x_j, y_j\}) = \varnothing$, $i = 1, 2, \ldots, n$;
(iv) every $z$ in $V_i - \{x_i, y_i\}$ belongs to an internal path in $G_i$ from $x_i$ to $y_i$;
(v) vertices $x_i$ and $y_i$ do not both belong to a cycle internal to $G_i$, $i = 1, 2, \ldots, n$.

Then the graph $G = (V, E)$ such that

$$V = \left(\bigcup_{i=1}^{n} V_i\right) \cup V_H$$

$$E = \left(\bigcup_{i=1}^{n} E_i\right) \cup E_H - \bigcup_{i=1}^{n} \{(x_i, y_i) \mid x_i, y_i \in V_i \text{ and } \mu(G_i) = (x_i, y_i)\}$$

is called the expanded graph of $H$ defined by family $\mathscr{G}$ and mapping $\mu$. Each $G_i$ is the expansion graph of the arc $(x_i, y_i)$ in $E_H$ such that $\mu(G_i) = (x_i, y_i)$ and every arc $(x_i, y_i) = \mu(G_i)$ is a macroarc. It follows from the definition of reduced and expanded graph that, if $G^{(R)}$ is the reduced graph associated with $G$ and defined by the family $\mathscr{G}$, then $G$ is the expanded graph of $G^{(R)}$ defined by $\mathscr{G}$ and by a mapping $\mu$ such that $\mu(G_i) = (x_i, y_i)$ where $G_i \in \mathscr{G}$, $(x_i, y_i) \in E^{(R)}$, $x_i \in I(G_i)$ and $y_i \in O(G_i)$.

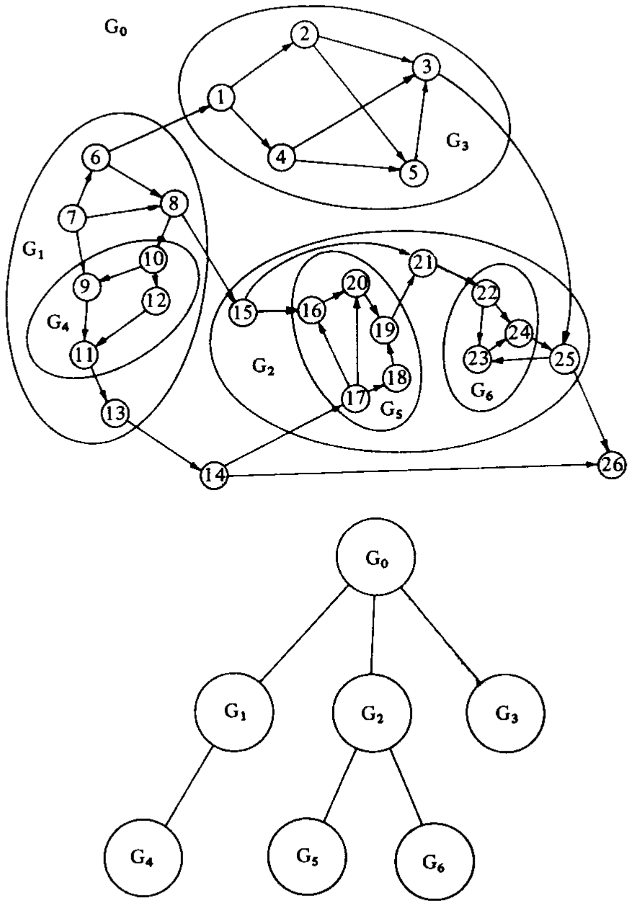


**Figure 1. A vertex-structured digraph and the tree structure describing it.**

As in the case of vertices, two operators, called arc expansion and arc reduction and denoted by $A_{\text{Exp}}$ and $A_{\text{Red}}$ respectively, are defined. $A_{\text{Exp}}$ maps the reduced graph $G^{(R)}$ and the family $\mathscr{G} = \{G_i\} i = 1, 2, \ldots, n$ into the expanded graph $G$, i.e.

$$A_{\text{Exp}}(G^{(R)}, \mathscr{G}, \mu) = G.$$

Conversely, $A_{\text{Red}}$ maps $G$ into $G^{(R)}$ as follows

$$A_{\text{Red}}(G, \mathscr{G}) = (G^{(R)}, \mu)$$

where $\mu$ is defined as above.

A structured graph is then defined as a pair $\gamma = (\mathscr{G}, T)$ where

$\mathscr{G} = \{G_i \mid i = 1, 2, \ldots, n\}$ is a family of graphs,

$T$ is a rooted tree with $n$ nodes, and there exists a one-to-one mapping $\tau$, which maps each node $t_i$ of $T$ into a unique graph $G_i$ in $\mathscr{G}$ (i.e. $\tau(t_i) = G_i$, $t_i \in T$ and $G_i \in \mathscr{G}$) such that every graph $G_j$ mapped by $\tau^{-1}$ into a child $t_j$ of a non-terminal node $t_i$ of $T$ is the expansion graph of a macroelement (macrovertex or macroarc) belonging to the graph $G_i = \tau(t_i)$, $G_i \in \mathscr{G}$.

A structured graph is called vertex-structured if it contains only macrovertices, and arc-structured if it contains only macroarcs. The graphs $G_i$ belonging to the family $\mathscr{G}$ are called graph components of $\gamma$. The graph

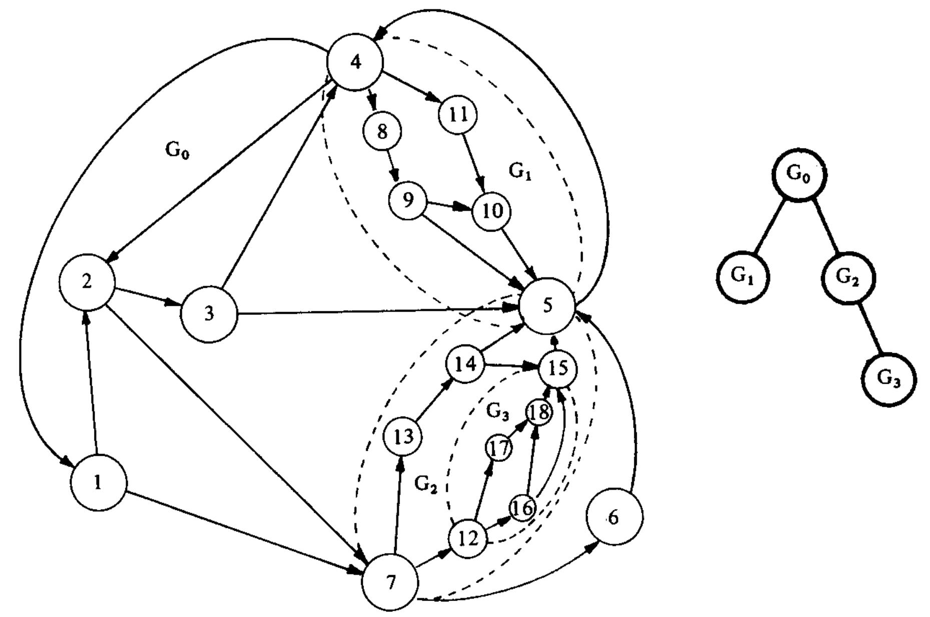


**Figure 2. An arc-structured digraph and the tree structure describing it.**

component associated with the root of $T$ is called the root graph of $\gamma$ and is denoted as $G_\gamma$. Figs 1 and 2 show respectively an example of a vertex-structured and of an arc-structured graph together with the tree structure describing them.

A structured graph $\gamma$ can be regarded as a variable-resolution representation of a graph, called the completely expanded graph, defined by $\gamma$ and denoted by $G_\gamma = (V_\gamma, E\gamma)$. $G_\gamma$ is defined in terms of operator recursive expansion, described below.

From the expansion operators $V_{\text{Exp}}$ and $A_{\text{Exp}}$ introduced before, we can define an expansion operator Exp, which is independent of the kind of graphs under consideration. To this aim, we denote the family of the expansion graphs of $G_i$ in $\gamma$ by $\mathscr{G}_i$, that is

$$\mathscr{G}_i = \{G_k \in \mathscr{G} \mid G_k \text{ is a child graph of } G_i \in \mathscr{G}\}$$

and we define the extended family of the expansion graphs of $G_i$ in $\gamma$, denoted by $\overline{\mathscr{G}}_i$, as follows:

if $\gamma$ is an arc-structured graph, then $\overline{\mathscr{G}}_i = \{(G_j, \mu(G_j)) \mid G_j \in \mathscr{G}_i\}$;

if $\gamma$ is a vertex-structured graph, then $\overline{\mathscr{G}}_i = \{(G_j, \mu(G_j), \phi_j, \psi_j) \mid G_j \in \mathscr{G}_i\}$. Hence, operator Exp is defined over $G_i \in \mathscr{G}_i$ as

$$\operatorname{Exp}(G_i, \overline{\mathscr{G}}_i) = \begin{cases} G_i \text{ if } \mathscr{G}_i = \varnothing \\ \overline{G}_i \text{ otherwise} \end{cases}$$

where $\overline{G}_i$ is the expanded graph of $G_i$ defined by the extended family $\overline{\mathscr{G}}_i$. The recursive expansion operator RExp is then expressed as

$$\operatorname{RExp}(G_i, \overline{\mathscr{G}}_i) = \begin{cases} G_i \text{ if } \mathscr{G}_i = \varnothing \\ \operatorname{RExp}(\operatorname{Exp}(G_i, \overline{\mathscr{G}}_i), \overline{\mathscr{G}}_i^*) \text{ otherwise} \end{cases}$$

where

$\overline{\mathscr{G}}_i^*$ is the extended family defined by $\overline{\mathscr{G}}_i^* = \bigcup_{G_j \in \mathscr{G}_i}$ $\{\mathscr{G}_m \mid \mathscr{G}_m$ is the family of expansion graphs of the macroelements of $G_i \in \mathscr{G}_i\}$ and $\overline{\mathscr{G}}_i$ is the extended family associated with $\mathscr{G}_i$.

Let $G_r$ in $\mathscr{G}$ be the root graph of $\gamma$. Then the completely expanded graph defined by $\gamma$ is given by

$$G_\gamma = \operatorname{RExp}(G_r, \overline{\mathscr{G}}_r),$$

i.e. $G_\gamma$ is obtained by starting from $G_r$ and recursively replacing its macroelements with their expansion graphs.

In the remainder of this section, we introduce some concepts and definitions which are used in the next two sections.

A subgraph of a structured graph is defined by suitably combining the concepts of subtree and subgraph.

Let $T$ be a rooted tree. Then any connected subgraph of $T$ is called a subtree of $T$. A subtree $T'$ of $T$ is called a complete subtree of $T$ if and only if all descendants of the root of $T'$ in $T$ are nodes of $T'$. Let $\gamma = (\mathscr{G}, T)$ be a structured graph and $T'$ a subtree of $T$. Then $T'$ uniquely defined a subfamily $\mathscr{G}' \subset \mathscr{G}$ as follows

$$\mathscr{G}' = \{G_i \mid \tau(t_i) = G_i \text{ and } t_i \in T'\}.$$

The pair $\gamma' = (\mathscr{G}', T')$ is said to be the subgraph of $\gamma$ defined by $T'$. A more general definition of subgraph of a structured graph can be given by taking subgraphs of its graph components into consideration.

Let $\gamma = (\mathscr{G}, T)$ and $\eta = (\mathscr{H}, T')$ be two structured graphs such that $T'$ is a subtree of $T$. Let $G_i \in \mathscr{G}$ and $H_i \in \mathscr{H}$ be two graph components of $\gamma$ and $\eta$, respectively, corresponding to the same node $t_i$ of $T'$. Then $\eta$ is a subgraph of $\gamma$ if and only if

(i) $H_i \subset G_i$

(ii) if $t_i$ is not the root of $T'$, then $G_i$ and $H_i$ are both expansion graphs in $\gamma$ and $\eta$ respectively of a macroelement $e_i$, which belongs to $H_j \in \mathscr{H}$ (and thus to $G_j$), where $H_j$ and $G_j$ are the parent graphs of $H_i$ and $G_i$, respectively, in $\eta$ and $\gamma$ (i.e. $\mu(H_i) = \mu(G_i) = e_i$).

(iii) if $\gamma$ and $\eta$ are vertex-structured graphs, then, let $\phi_i$, $\psi_i$ and $\phi_i'$, $\psi_i'$ denote the association mappings defined for $G_i$ and $H_i$, respectively. If $(x, e_i) \in E_i'$ then $\phi_i'(x, e_i) = \phi_i(x, e_i) \in I(H_i)$ where $x \in V_j$. If $(e_i, y) \in E_i'$ then $\psi_i'(e_i, y) = \psi_i(e_i, y) \in O(H_i)$ where $y \in V_j$.

If $\eta$ is a subgraph of $\gamma$, then $\eta \subset \gamma$.

If $T'$ is a complete subtree of $T$, then $\eta$ is called a complete subgraph of $\gamma$. If $u$ is a macroelement of $\gamma$ and $G_i$ its expansion graph, then the complete subgraph $\gamma_i = (\mathscr{G}_i, T_i)$ defined by the subtree $T_i$ of $T$ whose root corresponds to $G_i$ is called the structured expansion graph of $u$ in $\gamma$. Then the graph $G_{\gamma_i} = \text{RExp}(G_i, \mathscr{G}_i)$ is called the complete expansion graph of $u$. Also, $G_{\gamma_i} \subset G_\gamma$.

Let $H \subset G_i$ be a subgraph of a graph component $G_i$ of $\gamma$. The subgraph of $\gamma$ induced by $H$, denoted by $\eta = (\mathscr{G}_H, T_H)$, is defined as follows. If $t_i$ is the node of $T$ associated with $G_i$, then $t_i$ is the root of $T_H$. Every $T_j \subset T$, subtree of $T$, rooted at a child node of $t_i$, such that $G_j = \tau(t_j)$ is the expansion graph of a macroelement of $G_i$ belonging to $H$, is a subtree of $T_H$ rooted at a child node of $t_i$. And

$$\mathscr{G}_H = \{H\} \cup \{G_k \in \mathscr{G} \mid G_k \neq G_i \text{ and } G_k = \tau(t_k),\ t_k \in T_H\}.$$

Given any subgraph of the completely expanded graph $G_\gamma$, we define the concept of projection of such a subgraph on any component of $\gamma$. Let $H = (V_H, E_H)$ and $K = (V_K, F_K)$ be two graphs. The intersection of $H$ and $K$ is the graph $M = (V_M, E_M)$ such that $V_M = V_H \cap V_K$ and $E_M = E_H \cap E_K$. Let $\gamma = (\mathscr{G}, T)$ be a structured graph and $H$ a subgraph of $G_\gamma$. Then the projection of $H$ on $G_i$ is the smallest subgraph $H_i$ of $G_i$ such that

$$G_{\gamma_i} \cap H \subseteq H_{\eta_i},$$

where $H_{\eta_i}$ is the completely expanded graph associated with the subgraph $\eta_i \subset \gamma$ induced by $H$. The following properties follow immediately from the definition of projection.

(i) The projection of $H \subset G_\gamma$ on any $G_i$ of $\gamma$ is unique.

(ii) The projection of $H$ on the root graph of $\gamma$ is not empty if and only if $H$ is not empty.

(iii) The projection of $G_\gamma$ on any $G_i \in \mathscr{G}$ is the same as $G_i$.

The connectivity properties of a structured graph $\gamma$ are closely related to those of its corresponding completely expanded graph $G_\gamma$. A structured graph $\gamma = (\mathscr{G}, T)$ is said to be connected if and only if each graph component of $\gamma$ is connected. From the definition of arc-structured graph, we have that an arc-structured graph is connected if and only if its root graph is connected. It can easily be proved that $G_\gamma$ is connected if and only if $\gamma$ is connected.

Let $\gamma = (\mathscr{G}, T)$ be a structured digraph. Then $\gamma$ is said to be strongly connected if and only if every $G_i \in \mathscr{G}$ is strongly connected. The concept of strong connectivity does not make sense when referred to arc-structured digraphs, since any component which is not the root graph cannot be strongly connected because of the constraints imposed by the definition of arc reducibility. The following results are a direct consequence of the definitions of strongly connected graph and expansion.

**Lemma 2.1.** Let $\gamma = (\mathscr{G}, T)$ be a vertex-structured digraph. If $\gamma$ is strongly connected, then $G_\gamma$ is strongly connected.

The reverse is not true, as shown by the example depicted in Fig. 3.

**Lemma 2.2.** Let $\gamma = (\mathscr{G}, T)$ be a strongly connected vertex-structured digraph. Then

(i) any subgraph $\gamma' = (\mathscr{G}', T')$ of $\gamma$ defined by a subtree $T'$ of $T$ is strongly connected

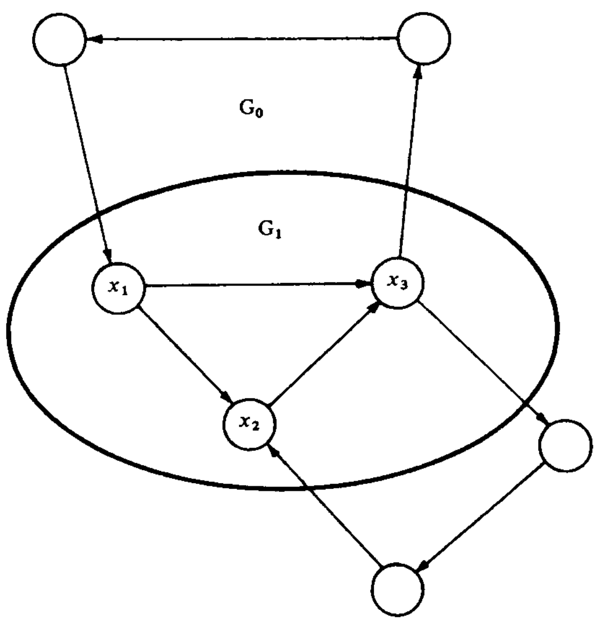


**Figure 3. An example of a non-strongly connected vertex-structured digraph $\gamma$ such that its corresponding completely expanded graph is strongly connected: $\gamma$ is not strongly connected since $G_1$ is not strongly connected.**

(ii) the structured and the complete expansion graphs of any macrovertex $u$ of $\gamma$ are strongly connected.

## 3. PATHS IN A STRUCTURED GRAPH

In this section we extend the concept of path to structured graphs. Since a structured graph is defined as a hierarchy of graphs, it is natural to define a path in such a structure as a collection of paths, organised in a similar tree to the given structured graph.

A path $P_{xy}$ between a pair of vertices $x$ and $y$ of a graph $G$ is defined as an ordered sequence of vertices of $G$:

$$P_{xy} = [z_1, z_2, \ldots, z_m] \quad \text{such that} \quad z_1 = x, z_m = y$$
$$\text{and} \quad (z_i, z_{i+1}),\ 1 \leqslant i \leqslant m-1 \text{ is an arc of } G.$$

Let $\gamma = (\mathscr{G}, T)$ be a vertex-structured graph, $G_i = (V_i, E_i)$ and $G_j = (V_j, E_j)$, $i \neq j$, be two graphs in $\mathscr{G}$ and $x$, $y$ two vertices, $x \in V_i$ and $y \in V_j$. Then, a path $P_{xy}$ in $\gamma$ between $x$ and $y$, $x \neq y$, is a finite sequence of paths $P_{xy} = [P_{x_1 y_1}, P_{x_2 y_2}, \ldots, P_{x_m y_m}]$, with $x_1 = x$, $y_m = y$, and $y_i = x_{i+1}$, $i = 1, 2, \ldots, m-1$, such that

(i) Every $P_{x_k y_k}$, $1 \leqslant k \leqslant m$, is a simple path between a pair of vertices $x_k$ and $y_k$ in a graph $G_p \in G$.

(ii) If $P_{x_k y_k}$ is a path in $G_p$ and $P_{x_{k+1} y_{k+1}}$ is a path in $G_q$, then $G_p$ is either a direct descendant or a direct ancestor of $G_q$.

(iii) If $P_{x_k y_k}$ and $P_{x_e y_e}$, $k \leqslant e$, are both paths in $G_p$, then every path $P_{x_s y_s}$, $k+1 \leqslant s \leqslant e-1$, is a path in a graph $G_q$ which must be either a descendant of $G_p$ or $G_p$ itself.

(iv) For every path $P_{x_k y_k}$ in a graph $G_p \in \mathscr{G}$ we have that if $k = 1$, $x_k = x$. Otherwise, let $P_{x_{k-1} y_{k-1}}$ be a path in $G_q$. If $G_q$ is the parent or an ancestor graph of $G_p$, then $x_k$ is the boundary vertex of $G_p$ associated with the last arc of $P_{x_{k-1} y_{k-1}}$. If $G_q$ is a child or a descendant graph of $G_p$, then $x_k$ is the macrovertex expanded into $G_q$ or into an ancestor of $G_q$.

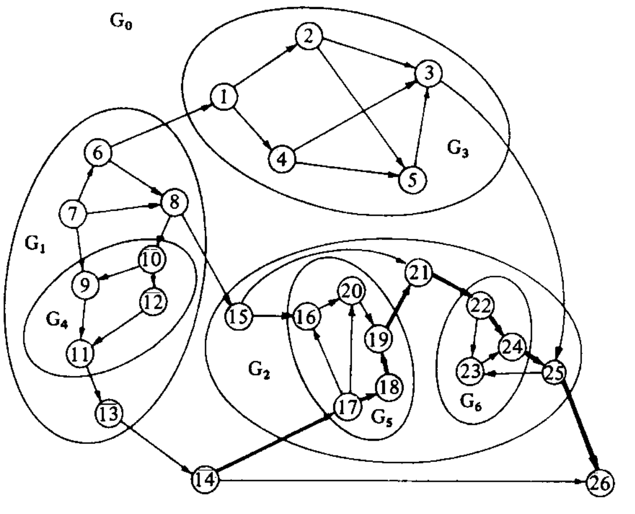


**Figure 4.** $P = [14, 17, 18, 19, 21, 22, 24, 25, 26]$ **is an example of a path in the vertex-structured digraph depicted in Fig. 1.**

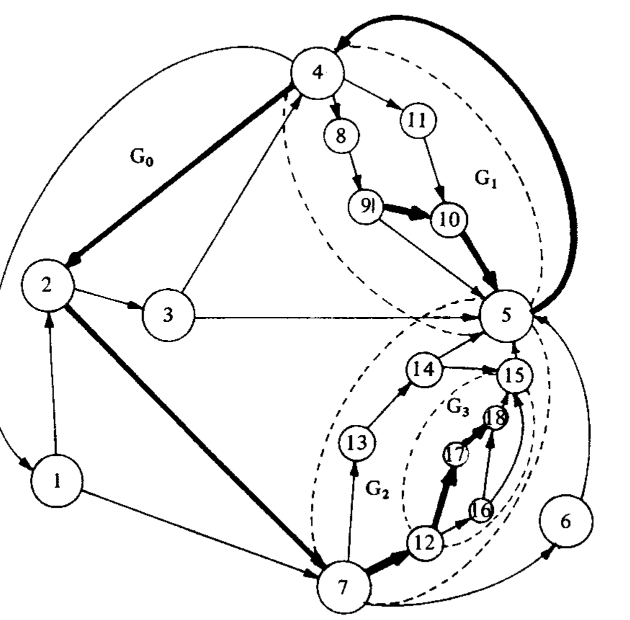


**Figure 5.** $P = [9, 10, 5, 4, 2, 7, 12, 17, 18]$ **is an example of a path in the arc-structured digraph depicted in Fig. 2.**

Similarly, if $k = m$, then $y_k = y$. Otherwise, let $P_{x_{k+1}y_{k+1}}$ be a path in $G_q$. If $G_q$ is the parent or an ancestor graph of $G_p$, then $x_k$ is the macrovertex expanded into $G_q$; if $G_q$ is a child or a descendant graph of $G_p$, then $x_p$ is the boundary vertex of $G_p$ associated with the first arc of $P_{x_{k+1}y_{k+1}}$.

Fig. 4 shows an example of path in the vertex-structured graph depicted in Fig. 1.

The definition of path in an arc-structured graph is simpler because the association between a macroarc $u$ and its expansion graph is completely defined by the two boundary vertices which belong to both the graph component containing macroarc $u$ and the expansion graph of $u$.

Let $\gamma = (\mathscr{G}, T)$ be an arc-structured graph, $G_i = (V_i, E_i)$ and $G_j = (V_j, E_j)$, $i \neq j$, be two components of $\gamma$, and $u$ and $v$ two vertices of $\gamma$, $u \in V_i$ and $v \in V_j$. Then a path $P_{uv}$ in $\gamma$ is a finite sequence of paths $P_{u_k v_k}$ defined as $P_{uv} = [P_{u_1 v_1}, P_{u_2 v_2}, \ldots, P_{u_n v_n}]$ with $u = u_1$, $v = v_n$, and $v_i = u_{i-1}$, $i = 2, \ldots, n$ such that

(i) $P_{u_k v_k}$, $k = 1, 2, \ldots, n$ is a simple path from $u_k$ to $v_k$ in a graph component $G_p$.

(ii) If $P_{u_k v_k}$ is a path in $G_p$ and $P_{u_{k+1} v_{k+1}}$ is a path in $G_q$, $G_q \in \mathscr{G}$, then $G_p$ is either a direct ancestor or a direct descendant of $G_q$.

(iii) If $P_{u_k v_k}$ and $P_{u_e v_e}$ are both paths in $G_p$, then every path $P_{u_s v_s}$, $k+1 \leqslant s \leqslant e-1$ is a path in a graph $G_q$, which must be either a descendant of $G_p$ or $G_p$ itself.

It follows immediately from the previous definition that if $P_{u_k v_k}$, $P_{u_{k+1} v_{k+1}}$ and $P_{u_{k-1} v_{k-1}}$ are paths of the sequence $P_{uv}$ belonging to $G_p$, $G_q$ and $G_s \in \mathscr{G}$, respectively, we have that:

(i) if $G_q$ is the parent graph of $G_p$, then $v_k$ is a boundary vertex of $G_p$; otherwise $u_{k+1}$ is the input vertex of $G_q$;

(ii) if $G_s$ is the parent graph of $G_p$, $u_k$ is the input vertex of $G_p$; otherwise $v_{k-1}$ is a boundary vertex of $G_s$.

Fig. 5 shows an example of a path in the arc-structured digraph depicted in Fig. 2.

## 4. FUNDAMENTAL PROPERTIES OF PATHS

Most properties presented in this section hold for both vertex- and arc-structured graphs.

First we introduce the definition of restriction of a path in a structured graph to a graph component $G_k$ of the structure. Let $\gamma = (\mathscr{G}, T)$ be a structured graph, and $P_{uv}$ be a path in $\gamma$ between a pair of vertices $u$ and $v$. For the sake of brevity, we denote $P_{uv}$ as $P = [P_1, P_2, \ldots, P_m]$ where $P_i = P_{u_i v_i}$, $i = 1, 2, \ldots, m$. The restriction $P^{(k)}$ of $P$ to a graph component $G_k$ is a simple path, which is either empty, if no element $P_i$ is a path in $G_k$, or is defined as follows.

Let $\bar{P} = [P_{j_1}, P_{j_2}, \ldots, P_{j_q}]$ be an ordered subsequence of $P$ such that

(i) every $P_{j_d}$, $d = 1, 2, \ldots, q$, is a path in $G_k$;

(ii) every $P_i$ belonging to $P$, but not to $\bar{P}$ is not a path in $G_k$.

Then $P^{(k)} = \bigcup_{d=1}^{q} P_{j_d}$.

Note that the union of two paths $P_1 = [x_1, \ldots, x_k]$ and $P_2 = [y_1, \ldots, y_n]$ belonging to a graph $H$ such that $x_k = y_1$ is defined as the path $P_3 = [x_1, \ldots, x_k, y_2, \ldots, y_n]$. From the above definition it follows immediately that every $P_{j_d}$ in $P$ is a path $P_{x_{j_d} y_{j_d}}$, where $x_{j_d}$ and $y_{j_d}$ are vertices of the graph $G_k$ to which $P_{j_d}$ belongs. Also, $P$ is a simple path between a pair of boundary vertices in $G_k$.

**Lemma 4.1.** Let $\gamma = (\mathscr{G}, T)$ be a structured graph, $P$ a path in $\gamma$ and $\mathscr{G}' = \{G_k \in \mathscr{G} \mid$ there exists at least one path $P_i$ in $P$ which belongs to $G_k\}$ be a subfamily of $\mathscr{G}$. Then, $\bar{T} = \{t \in T \mid \tau(t)$ is an element of $\mathscr{G}'\}$ is a subtree of $T$.

**Proof.** It follows from condition (ii) in the definition of path in a structured graph.

**Theorem 4.1.** Let $\mathscr{P}$ be a family of paths defined as: $\mathscr{P} = \{P^{(k)} \mid P^{(k)}$ is the non-empty restriction of a path $P$ in $\gamma$ to a graph $G_k \in \mathscr{G}\}$. Then $\pi = (\mathscr{P}, \bar{T})$ is a subgraph of $\gamma$.

**Proof.** For every $P^{(k)} \in \mathscr{P}$ there exists exactly one $G_k \in \mathscr{G}$ such that $P^{(k)}$ is a subgraph of $G_k$. If $G_k$ is not the root graph, let $G_j$ be the parent graph of $G_k$ and $u$ the

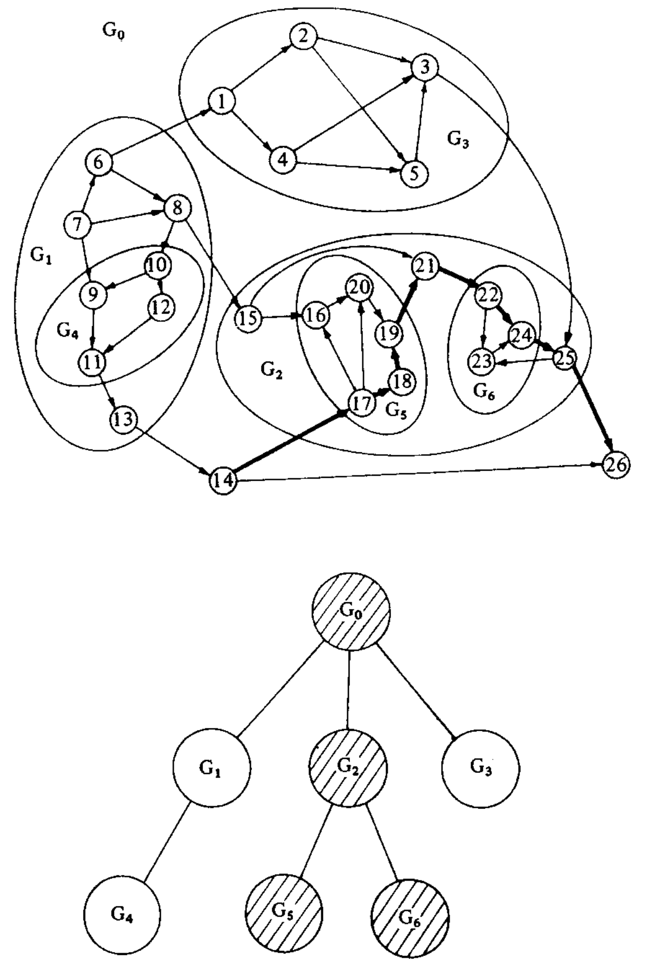


**Figure 6. Subgraph of the vertex-structured digraph of Fig. 1 associated with the path $P$ in $\gamma$ shown in Fig. 4.**

macroelement of $G_j$ expanded into $G_k$. Then $P^{(k)}$ is a reducible component in the graph obtained from $G_j$ by replacing $u$ with $P^{(k)}$, since it is a simple path. Finally, we have to prove that, if $P^{(k)} \in \mathscr{P}$ is not the root graph of $P$, $P^{(k)}$ is the expansion graph of a macroelement belonging to the restriction of $P$, say $P^{(j)}$, to $G_j$. Since $P^{(k)} = \bigcup_{d=1}^{s} P_{j_d}$, let $P_g$ be the element in $P$ which is the immediate predecessor of $P_{j_s}$ in $P$ ($P_g$ must exist since $P^{(k)}$ does not correspond to the root of $T$); then $P_g = [x_1, x_2, \ldots, x_g]$. Thus, if $\gamma$ is vertex-structured, $x_g$ is the macrovertex of the parent graph $G_j$ of $G_k$ which is expanded into $G_k$. If $\gamma$ is arc-structured, there exists a macroarc $(x_g, x_j)$ in $G_j$ expanded into $P^{(k)}$. Hence the theorem.

Subgraph $\pi$ as defined in the statement of Theorem 4.1 is called the subgraph of $\gamma$ associated with $P$. Fig. 6 shows the subgraph of the vertex-structured digraph of Fig. 1 associated with path $P$ depicted in Fig. 4.

**Lemma 4.2.** Let $P = [P_1, P_2, \ldots, P_m]$ be a path in a structured graph $\gamma = (\mathscr{G}, T)$. Then every subsequence $P' = [P_i, P_{i+1}, \ldots, P_j]$, $i \leqslant j$, such that every $P_k$ in $P'$, $i \leqslant k \leqslant j$ is an element of $P$, is a path in $\gamma$.

Thus, every subsequence $P'$ of $P$, $P' = [P_i, \ldots, P_j]$ defined as in Lemma 4.2 is called a subpath of $P$.

In the remainder of this section we examine the relationship between the general paths in the completely expanded graph $G_\gamma$ associated with a structured graph $\gamma$ and those obtained by expansion of a path in $\gamma$. A path $Q$ in $G_\gamma$ is said to be structured if and only if the projection of $Q$ on any $G_k \in \mathscr{G}$ is either empty or a simple path. On the other hand, a path in $G_\gamma$ is called a canonical path if and only if it does not necessarily satisfy the structuredness property stated above.

**Lemma 4.3.** Let $G_\pi$ be the completely expanded graph associated with the subgraph $\pi = (\mathscr{P}, \bar{T})$ of $\gamma$ defined by a path $P$ in $\gamma$. Let $\bar{\gamma} = (\bar{\mathscr{G}}, \bar{T})$ be the subgraph of $\gamma$ defined by $\bar{T}$ and $G_{\bar{\gamma}}$ the completely expanded graph associated with $\bar{\gamma}$. Then $G_\pi$ is a path in $G_{\bar{\gamma}}$.

**Proof.** Since $\pi$ and $\bar{\gamma}$ are defined by the same tree structure, $G_\pi$ is a subgraph of $G_{\bar{\gamma}}$. Furthermore, $G_\pi$ is a path in $G_{\bar{\gamma}}$, since $G_\pi$ is obtained from $\pi$ by recursively substituting every macroelement with a path. Hence the lemma.

**Theorem 4.2.** Let $\pi$ and $\bar{\gamma}$ be defined as in Lemma 4.3. Then $G_\pi$ is a structured path in $G_{\bar{\gamma}}$.

**Proof.** It follows immediately by observing that the projection of $G_\pi$ on any $G_k \in \bar{\mathscr{G}}$ is a path $P^{(k)} \in \mathscr{P}$, since the projection of a subgraph $H_i$ of a component $G_i$ of a structured graph on $G_i$ is the same as $H_i$. Hence the theorem.

**Corollary 4.1.** Let $P$ be a path in a structured graph $\gamma$ such that the subgraph $\pi = (\mathscr{P}, \bar{T})$ of $\gamma$ associated with $P$ is a complete subgraph of $\gamma$. Then $G_\pi$ is a structured path in $G_\gamma$.

**Proof.** Let $\bar{\gamma} = (\bar{\mathscr{G}}, \bar{T})$ be the subgraph of $\gamma$ defined by $\bar{T}$. Since $\pi$ is a complete subgraph of $\gamma$, then also $\bar{\gamma}$ is a complete subgraph of $\gamma$ as well. Thus $G_{\bar{\gamma}}$ is a subgraph of $G_\gamma$. Since $G_\pi$ is a structured path in $G_{\bar{\gamma}}$, $G_\pi$ is a structured path in $G_\gamma$. Hence the corollary.

**Lemma 4.4.** Let $\pi = (\mathscr{P}, \bar{T})$ be a subgraph of a structured graph $\gamma$ such that every $P^{(k)} \in \mathscr{P}$ is a path. Then there exists a path $P$ in $\gamma$ such that $\pi$ is the subgraph of $\gamma$ associated with $P$.

**Proof.** We give a constructive proof of the lemma based on algorithm PATH described below, which builds up a sequence of paths from the subgraph $\pi$ of $\gamma$ defined as above.

```
Algorithm PATH ($P^{(k)}$, $P_k$)
  // $P^{(k)}$ denotes a component of $\pi$;
     $P_k$ is the output sequence of paths //
  $P_k \leftarrow \varnothing$;
  $P^* \leftarrow P^{(k)}$;
  let $u$ be the first macroelement in $P^{(k)}$;
  while there exists a $u$ do
     split $P^*$ into two subsequences, say $P'_k$ and $P''_k$, such
     that $P'_k$ is the subpath of $P^*$ having $u$ as last vertex
     and $P''_k$ the subpath of $P^*$ having $u$ as first vertex;
     $P^* \leftarrow P''_k$;
     $P_k \leftarrow P'_k \cup P_k$;
     let $P^{(j)}$ be the expansion graph of $u$ in $\pi$;
     PATH ($P^{(j)}$, $P_j$);
     $u \leftarrow$ next macroelement of $P^{(k)}$;
     $P_k \leftarrow P_k \cup P_j$;
  end while;
  $P_k \leftarrow P_k \cup P^*$
end PATH.
```

We prove now that the sequence $P$ obtained by applying algorithm PATH to the root graph of $\pi$ is a path in $\gamma$ according to the definition of path in a structured graph. Condition (i) is satisfied since every element of $P$ is a subpath of some $P^{(k)} \in \mathscr{P}$ and, therefore, is a path in the graph component $G_k$ of $\gamma$ associated with that node of $\bar{T}$ and $T$ which corresponds to $P^{(k)}$ in $\pi$. Conditions (ii) and (iii) follow immediately because algorithm PATH is recursively activated every time a macroelement in any component $P^{(k)}$ is encountered. To prove that condition (iv) holds for a vertex-structured graph, we consider an element $P_j \in P$, $P_j = P_{x_j y_j}$, i.e. a path between vertices $x_j$, $y_j$ in the graph $G_p \in \mathscr{G}$. If $j = 1$, then $x_j$ must be a simple vertex, since $P_j$ is the first path of the sequence. Otherwise (if $j > 1$), if $P_{j-1}$ is a path in a parent or ancestor graph of $G_p$, then $x_j$ must be a boundary (or an input) vertex of $G_p$ because of the definition of subgraph and algorithm PATH. If $P_{j-1}$ is a path in a child or descendant graph of $G_p$, then $x_j$ must be a macrovertex for the same reasons. Similar arguments apply to prove that $y_j$ can be either a simple vertex, a macrovertex, or a boundary vertex. Hence the lemma.

**Lemma 4.5.** Let $\pi$ be a subgraph of a structured graph $\gamma$ defined as in the statement of Lemma 4.4. Then the path $P$ associated with $\pi$ is unique.

**Proof.** Suppose that another path $P'$ exists in $\gamma$, $P' \neq P$, such that $\pi$ is the subgraph of $\gamma$ associated with $P'$. From the definition of restriction of a path and Theorem 4.1, it follows that every $P^{(k)} \in \mathscr{P}$ is the restriction of both paths $P'$ and $P$ on a graph $G_k \in \mathscr{G}$. Thus both paths reduce to the same path on each $G_k \in \mathscr{G}$. Hence Lemma 4.5.

**Theorem 4.3.** Let $Q$ be a structured path in $G_\gamma$. Then, there exists a path $P$ in $\gamma$ such that the completely expanded graph $G_\pi$ associated with the subgraph $\pi$ of $\gamma$ associated with $P$ is the same as $Q$.

**Proof.** From the definition of structured path we have that the projection of $Q$ on any $G_k \in \mathscr{G}$ is either empty or is a simple path $Q_k$. Let

$$\mathscr{P} = \{Q_j \mid Q_j \text{ is the non-empty projection of } Q \text{ on } G_j \in \mathscr{G}\}$$

and

$$T - \{t_i \in T \mid \text{the projection of } Q \text{ on } G_i \in \mathscr{G} \text{ such that } \tau(G_i) = t_i \text{ is not empty}\}.$$

$\bar{T}$ is a subtree of $T$, since $Q$ would not be connected otherwise. Now the pair $\pi = (\mathscr{P}, \bar{T})$ is a subgraph of $G$ since

(i) Each $Q_j \in \mathscr{P}$ is a subgraph of a $G_j \in \mathscr{G}$.

(ii) Each $Q_j \in \mathscr{P}$ is reducible to a macroelement since a path satisfies the conditions of vertex and arc reducibility.

(iii) $Q_j \in \mathscr{P}$, different from the root of $\pi$, is the expansion graph of an element belonging to its parent graph. This follows from the definition of projection, because $Q$ is connected.

Thus $G_\pi$ is the same as $Q$ because of the definition of recursive expansion. Hence the theorem.

**Lemma 4.6.** The set of the structured paths in $G_\gamma$ between a pair of vertices $x$ and $y$ is a subset of the set of the canonical paths between the same pair.

**Lemma 4.7.** Let $P$ be a path in a structured graph $\gamma$ between a pair of vertices $x$ and $y$ in the same components $G_k \in \mathscr{G}$. Let $\pi = (\mathscr{P}, \bar{T})$ be the subgraph of $\gamma$ defined by $P$, then $\bar{T}$ is a subtree of $T$, the root of which corresponds to $G_k$.

**Lemma 4.8.** Let $P = [P_1, P_2, \ldots, P_m]$ be a path in a structured graph $\gamma$ and $\pi = (\mathscr{P}, \bar{T})$ the subgraph of $\gamma$ defined by $P$. Then, for every $G_j \in \mathscr{G}$, $G_j$ leaf graph of $\bar{\gamma} = (\bar{\mathscr{G}}, \bar{T})$, there exists only one $P_i \in P$ such that the restriction $P^{(j)}$ of $P$ to $G_j$ is the same as $P_i$.

**Proof.** It follows immediately by observing that each leaf graph $G_j$ does not contain any macroelement in $\gamma$.

**Theorem 4.6.** Let $\gamma = (\mathscr{G}, T)$ be a structured graph and $u$, $v$ two simple vertices of $G_\gamma$ belonging to the same component $G_k \in \mathscr{G}$. If a path between $u$ and $v$ exists in $G_k$, then there exists a structured path between them in $G_\gamma$.

**Proof.** Let $P_{uv}$ the path between $u$ and $v$ in $G_k$. Then a structured path $Q_{uv}$ between $u$ and $v$ in $G_\gamma$ can be constructed by starting from $P_{uv}$ and recursively replacing every macroelement belonging to $P_{uv}$ with a path between the corresponding boundary vertices, which must exist because of the definitions of vertex and arc reducibility. Hence the theorem.

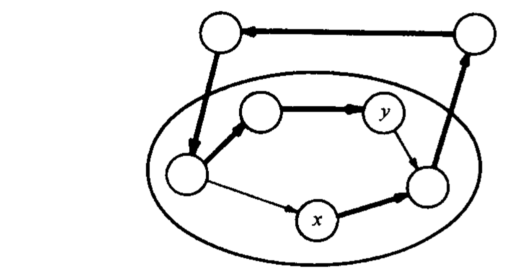
Algorithm BUILD_PATH described below constructs a path $P$ in $G_\gamma$ between a pair of vertices belonging to a component $G_k$ of a vertex-structured graph $\gamma$ from a path $P_k$ in $G_k$ between the same pair.

```
Algorithm BUILD_PATH ($P_k$, $G_k$, $P$)
    // $P_k$ is a path in a graph component $G_k$ of $\gamma$;
   $P$ is the resulting path in $G_\gamma$; at the first activation
   $P \leftarrow \emptyset$//
   $Q \leftarrow P_k$;
   let $w$ be the first macrovertex in $P_k$;
   while there exists a $w$ do
      let $a$ and $b$ be respectively the
      predecessor and successor of $w$ in $P_k$;
      let $G_j$ be the expansion graph of $w$ in $\gamma$;
      let $a_w$ and $b_w$ be respectively the
      boundary vertices of $G_j$ corresponding to $w$;
      split $Q$ into two subsequences $Q_1$ and
      $Q_2$ having $w$ as extreme vertex;
      $P \leftarrow P \cup Q_1$;
      let $P_j$ be a path between $a_w$ and $b_w$ in $G_j$;
      BUILD_PATH ($P_j$, $G_j$, $\bar{P}$);
      $P \leftarrow P \cup \bar{P}$;
      $Q \leftarrow Q_2$;
      let $w$ be the next macrovertex in $P_k$;
   end while;
   $P \leftarrow P \cup Q$;
end BUILD_PATH.
```

In the following two theorems we develop sufficient conditions on the existence of a structured path in $G_\gamma$.

**Theorem 4.7.** Let $\gamma = (\mathscr{G}, T)$ be a vertex-structured graph, $u$ and $v$ two simple vertices belonging to $G_k \in \mathscr{G}$ and $Q$ a path in $G_\gamma$ between vertices $u$ and $v$. If the projection of path $Q$ on a graph $G_j \in \mathscr{G}$ being not empty implies that either $G_j = G_k$ or $G_j$ is a descendant of $G_k$, then there exists a structured path in $G_\gamma$ between $u$ and $v$.

**Proof.** A structured path between $u$ and $v$ can be constructed from $Q$ as follows. For every $G_j \in \mathscr{G}$, $G_j \neq G_k$, such that the projection $Q_j$ of $Q$ on $G_j$ is not empty, let



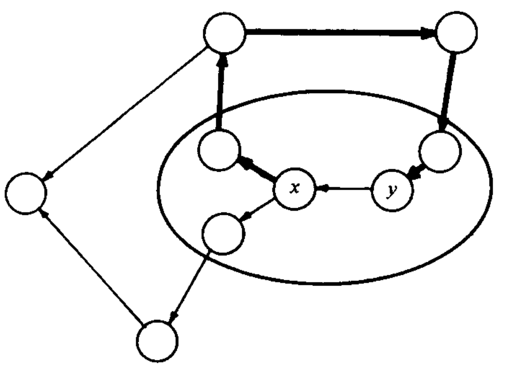


Figure 7. In both examples there exists a canonical path from $x$ to $y$ in the completely expanded graph, but no structured path joining them.

$u_j$ and $w_j$ denote the first and last vertex of $Q$, respectively, belonging to $G_j$. If $G_j$ does not contain either $u$ or $v$, then $u_j$ must be a boundary vertex (an input vertex if $\gamma$ is directed) and similarly $w_i$ must be a boundary vertex (an output vertex if $\gamma$ is a digraph). Thus the whole subpath of $Q$ between $u_j$ and $w_j$ (or the corresponding boundary vertices in their expansion graph) can be replaced with the expansion of an internal path connecting $u_j$ to $w_j$. Existence of such a path follows from the traversability property. Hence the theorem.

A stronger result for undirected arc-structured graphs holds, as shown in Lemma 4.9 below.

**Lemma 4.9.** Let $\gamma = (\mathcal{G}, T)$ be an undirected arc-structured graph and $Q$ a path between a pair of vertices $u$ and $v$ in $G_\gamma$ such that

(i) $u$ and $v$ belong to the same $G_k \in \mathcal{G}$;

(ii) the projection of $Q$ on a graph $G_j \in \mathcal{G}$ being not empty implies that $G_j$ is either the same as $G_k$ or a descendant of $G_k$.

Then $Q$ is a structured path in $G_\gamma$.

**Proof.** Suppose that $Q$ is not structured. Then the projection of $Q$ on the parent graph $G_i$ of $G_k$ or on one of the siblings of $G_i$ cannot be empty, which is a contradiction. Hence the lemma.

In general, however, the existence of a canonical path in $G_\gamma$ between a pair of vertices does not guarantee that there exists a structured path between the same pair, as shown by the two examples in Fig. 7.

The constructive proof of Theorem 4.7 suggests an algorithm which computes a structured path in the completely expanded graph $G_\gamma$ associated with a vertex-structured graph $\gamma$ from a given canonical path, which satisfies the conditions stated in Theorem 4.7.

```
Algorithm CP_TO_SP ($\gamma$, $P$)
// $\gamma = (G, T)$ is a vertex-structured graph and $P$ a
canonical path in $G_\gamma$ satisfying the conditions expressed
in Theorem 4.7//
  for every $G_j \in \mathcal{G}$ such that the projection of $P$ on $G_j$ is
  not empty or a simple path do
    let $u_j$ and $v_j$ be the first and last vertex respectively
    in $P_j$ which belong to $G_j$;
    let $P_j$ be the subpath of $P$ between $u_j$ and $v_j$;
    PATH1 ($G_j$, $u_j$, $v_j$, $P'_j$);
    replace $P_j$ with $P'_j$ in $P$;
  end for;
end CP_TO_SP.
```

```
Algorithm PATH1 ($G_i$, $u_i$, $v_i$, $P'$)
  $P' \leftarrow$ path in $G_i$ between $u_i$ and $v_i$;
  $w \leftarrow$ first macrovertex of $P'$;
  while there exists a $w$ do
    let $G_q$ be the first-level expansion graph of $w$ in $\gamma$;
    let $x_w$ and $y_w$ be the two boundary vertices of $G_q$
    corresponding to the predecessor $x$ and
    successor $y$, respectively, of $w$ in $P'$;
    PATH1 ($G_q$, $x_w$, $y_w$, $P''$);
    replace $w$ in $P'$ with $P''$;
    $w \leftarrow$ next macrovertex in $P'$;
  end while;
end PATH1.
```

In the case of undirected graphs, the existence conditions for structured paths are less restrictive, as established by the following lemma.

**Lemma 4.10.** Let $\gamma = (\mathcal{G}, T)$ be an undirected structured graph. Then there exists a structured path in $G_\gamma$ between any pair of vertices of $G_\gamma$.

**Proof.** Suppose that there exists in $G_\gamma$ a pair of vertices $v, w \in V_\gamma$ such that no structured path between $v$ and $w$ exists in $G_\gamma$. Since $G_\gamma$ is connected, then there exists a canonical path $Q + \bar{Q}_{uv}$ in $G_\gamma$, which is not structured. Thus, there exists $G_j \in \mathcal{G}$ such that the projection $Q_j$ of $Q$ on $G_j$ is not empty and also not a simple path. Two cases may occur: $Q_j$ is a cycle or $Q_j$ is the union of two or more disjoint paths. If $Q_j$ is a cycle, then consider the completely expanded graph defined by the complete subgraph $\gamma_j$ of $\gamma$ having root at $G_j$, say $G_{\gamma_j}$ and the restriction of $Q$ to $G_j$, say $\bar{Q}_{u_j v_j}$, which will be a path between two vertices $u_j$ and $v_j$ of $Q$. Then $\bar{Q}_{u_j v_j}$ should be replaced in $Q$ by a path $Q_{u_j v_j}$ in $G_{\gamma_j}$ such that $Q_{u_j v_j}$ can have a non-empty projection only on $G_j$ and on some of its descendants. If $Q_j$ is the union of two or more disjoint paths, then let $w_j$ and $z_j$ be the first and the last vertex, respectively, of $Q_j$. Replace the subpath of $Q$ between $w_j$ and $z_j$ in $G_\gamma$ with a path in $G_\gamma$ between the same pair which has a non-empty projection only on $G_j$ and on some or all of its descendants. Hence the lemma.

Acyclic structured graphs have some interesting properties related to the existence of structured paths in the completely expanded graph. Let $\gamma = (\mathcal{G}, T)$ be a structured graph. A path $P$ in $\gamma$ between a pair of vertices $u$ and $v$ is a cycle if and only if $u = v$. The result stated by Lemma 4.11 is an immediate consequence of the definitions of cycle and path in a structured graph.

**Lemma 4.11.** Let $C$ be a cycle in $\gamma$ and $\pi = (\mathscr{P}, \bar{T})$ the subgraph of $\gamma$ associated with $C$. Then the restriction $C^{(k)}$ of $C$ to any component $G_k$ of $\gamma$ is a cycle if and only if $G_k$ is the root graph of the graph $\bar{\gamma} = (\bar{\mathscr{G}}, \bar{T})$ defined by $\bar{T}$.

A structured graph $\gamma$ is said to be acyclic if and only if every $G_i \in \mathscr{G}$ is acyclic.

**Theorem 4.8.** Let $\gamma = (\mathscr{G}, T)$ be a structured graph and $G_\gamma$ be the corresponding completely expanded graph. The $\gamma$ is acyclic if and only if $G_\gamma$ is acyclic.

**Proof**

(1) *If* $\gamma$ is acyclic, then $G_\gamma$ is acyclic. Suppose that $G_\gamma$ contains a cycle. Two cases exist. First, all elements of the cycle belong to the same graph $G_j \in \mathscr{G}$, and thus $G_j$ contains a cycle, which is a contradiction. Secondly, there exist at least two graphs $G_i$ and $G_j \in \mathscr{G}$ such that the cycle contains elements of both $G_i$ and $G_j$. Then, let $G_k$ be the nearest common ancestor of $G_i$ and $G_j$ such that the restriction of the cycle to $G_k$ is not empty. But such a restriction is a cycle, which is a contradiction.

(2) If $G_\gamma$ is acyclic then $\gamma$ is acyclic. Suppose that $\gamma$ contains a cycle $C$. Then, at least one $G_i \in \mathscr{G}$ contains a cycle $C^{(i)}$. By recursive expansion of the macroelement in $C^{(i)}$, we obtain a cycle containing only simple elements, which is a cycle in $G_\gamma$. Thus we have a contradiction. Hence the theorem.

**Lemma 4.12.** Let $\gamma = (\mathscr{G}, T)$ be a structured graph. If $\gamma$ is acyclic, then (i) any subgraph of $\gamma$ is acyclic and (ii) any subgraph of $G_\gamma$ is acyclic.

**Lemma 4.13.** Let $\gamma$ be a vertex-structured or an undirected arc-structured graph. Then a path in $G_\gamma$ between a pair of distinct vertices is structured if and only if its projection on any $G_i \in \mathscr{G}$ contains no cycle.

**Proof.** We have only to prove that if the projection of a path $Q$ in $G_\gamma$ on any $G_i \in \mathscr{G}$ contains no cycle, then the path is structured. Suppose that $Q$ is not structured. This means that there exists $G_j \in G$ such that the projection $Q_j$ of $Q$ on $G_j$ is neither empty, nor a simple path. If $Q_j$ contains a cycle, we have a contradiction. Otherwise, if $Q_j$ is the union of two or more disjoint paths, then $G_j$ cannot be the root graph of $\gamma$, and also the projection of $Q$ on an ancestor of $G_j$ must contain a cycle involving a macroelement, which leads to a contradiction. Hence the lemma.

**Lemma 4.14.** Let $\gamma$ be a vertex-structured or an undirected arc-structured graph. Then, a cycle $C$ in $G_\gamma$ is structured if and only if there exists exactly one $G_k \in \mathscr{G}$ such that its projection on $G_k$ is a simple cycle and its projection on any other $G_i \in \mathscr{G}$, $G_i \neq G_k$, does not contain any cycle.

**Proof.** A proof similar to that of Lemma 4.13 can be easily constructed.

A structured graph $\gamma = (\mathscr{G}, T)$ is said to be weakly acyclic if and only if every $G_i \in \mathscr{G}$, which is not a leaf graph, is acyclic. It follows immediately that an acyclic structured graph is weakly acyclic.

**Theorem 4.9.** Let $\gamma = (\mathscr{G}, T)$ be a vertex-structured or an undirected arc-structured graph. Then, if $\gamma$ is weakly acyclic, every path in the completely expanded graph $G_\gamma$ is structured.

**Proof.** Suppose that there exists a path $P$ in $G_\gamma$ which is not structured. Thus a graph $G_i \in \mathscr{G}$ exists such that the projection $P_i$ of $P$ on $G_i$ is neither empty nor a simple path. Two cases occur.

(1) $P_i$ contains a cycle. Then, if $G_i$ is a leaf graph, $P_i$ is the same as $P$, and therefore $P$ is a cycle in a leaf graph. Thus $P$ is structured. Otherwise, we have a contradiction.

(2) $P_i$ is the union of two or more disjoint simple paths. Then, the projection of $P$ on an ancestor of $G_i$, say $G_j$, must contain a cycle, since a subpath of $P$ must exist between two boundary vertices of $G_i$ containing no vertex of $G_i$. Thus $G_j$ contains a cycle involving the macroelement expanded into $G_i$, which is a contradiction. Hence the theorem.

**Lemma 4.15.** Let $\gamma$ be a vertex-structured or an undirected arc-structured graph. Let $P$ be an unstructured simple path in $G_\gamma$. There exists at least one graph $G_i \in \mathscr{G}$ such that the projection of $P$ on $G_i$ contains a cycle involving a macroelement.

**Proof.** Suppose that such a graph $G_i$ does not exist. This means that the projection of $P$ on any graph $G_j \in \mathscr{G}$ must be empty, a simple path or a simple cycle containing no macroelement. In all these cases, $P$ is structured. Thus, we have a contradiction. Hence the lemma.

**Theorem 4.10.** Let $\gamma$ be a vertex-structured or an undirected arc-structured graph. Then every path on $G_\gamma$ is structured if and only if every $G_i \in \mathscr{G}$ does not contain any cycle involving macroelements.

**Proof.** If every path in $G_\gamma$ is structured, then no $G_i$ contains any cycle involving macroelements. Suppose that there exists $G_j \in \mathscr{G}$ which contains a cycle $C$ involving a macroelement $u$. An unstructured path $P$ exists in $G_\gamma$ between the two boundary simple vertices, say $w$ and $v$, belonging to the expansion graph $G_i$ of $u$ or to one descendant of $G_i$ and defined by the two arcs of $C$ incident in $u$. Such a path is obtained by recursively expanding the macroelements, different from $u$, which belong to $C$ and treating $w$ and $v$ as extreme vertices of $P$. Therefore, we have a contradiction. If every $G_i \in \mathscr{G}$ does not contain any cycle involving macroelements, then every path in $G_\gamma$ is structured. If there exists an unstructured path in $G_\gamma$, we have a contradiction because of the result stated by Lemma 4.15. Hence the theorem.

In the case of directed arc-structured graphs more restrictive conditions must be imposed to ensure that every path in $G_\gamma$ is structured.

**Theorem 4.11.** Let $\gamma = (\mathscr{G}, T)$ be an arc-structured graph. Then, if every $G_i \in \mathscr{G}$ satisfies the following condition:

for every macroarc $(x, y)$ in $G_i$ there does not exist any path in $G_i$ joining $x$ to $y$ or $y$ to $x$,

then every path in $G_\gamma$ is structured.

**Proof.** Suppose that an unstructured simple path $P$ exists in $G_\gamma$. Then there must exist a graph $G_j \in \mathscr{G}$ such that the projection of $P$ on $G_j$ is the union of two disjoint paths $P_{ux_j}$ (or $P_{x_j u}$) and $P_{y_j v}$ (or $P_{vy_j}$), where $x_j$ and $y_j$ denote, respectively, the input and output vertex of $G_j$, and the restriction of $P$ to the parent graph $G_k$ of $G_j$ is a path from $x_j$ to $y_j$ (or from $y_j$ to $x_j$). Therefore, $P$ does not satisfy the condition stated in the theorem, which is a contradiction. Hence the theorem.

Let $\gamma = (\mathscr{G}, T)$ be a structured graph and $c$ a cost function defined over the arcs of $\gamma$. Then a path of minimum cost in $\gamma$ is said to be a shortest path in $\gamma$. Similarly, a structured path in $G_\gamma$ of minimum cost is called a structured shortest path. Let CSP denote a canonical shortest path in $G_\gamma$ between a pair of vertices and SSP a structured shortest path between the same pair. Then

$c(\text{CSP}) \leqslant c(\text{SSP})$ in $G_\gamma$

**Corollary 4.2.** Let $\gamma = (\mathscr{G}, T)$ be a vertex-structured (or an undirected arc-structured) graph. If $\gamma$ satisfies the condition stated in Theorem 4.10 then a shortest path in $G_\gamma$ is structured.

**Corollary 4.3.** Let $\gamma = (\mathscr{G}, T)$ be an arc-structured graph. If $\gamma$ satisfies the condition stated in Theorem 4.11, then the shortest path between any pair of vertices of $G_\gamma$ is structured.

## 5. CONCLUSION

In this paper we study the properties of paths in a structured graph. The concept of path in a structured graph is defined and the properties of paths are investigated. Since a structured graph provides a representation of a graph model at variable detail levels, the relationships between structured paths and the canonical ones in the graphs represented by a structured graph are studied in detail. Algorithms to construct a path in a structured graph, and to compute a structured path from a given canonical one, are presented.

Future developments of this work include an implementation of algorithms for finding structured paths in directed graphs at any level of detail within a system for graph manipulation based on vertex-structured digraphs with specific applications in the design and analysis of computer networks and VLSI systems.

**Acknowledgements**

This work has been supported by the Italian National Research Council. The authors are grateful to Sandra Burlando for her help in typing this manuscript.

## REFERENCES

1. M. Ancona and L. De Floriani, Structured graphs and their applications, *Proc. Second IASTED Applied Modeling and Simulation Symp., Paris, 1982*, 155–159.
2. M. Ancona and L. De Floriani, Computational algorithms for hierarchically structured project networks; *Operations Research Letters*, **1** (5), 170–176 (1982).
3. M. Ancona, L. De Floriani, O. Trebino and A. Zamana, A system for structured graph definition, *Rivista di Informatica* **14** (4), 361–377 (1984) (in Italian).
4. M. Ancona, L. De Floriani and J. S. Deogun, *Structured Graphs: Formalism, Properties and Algorithms*, Technical Report no. 158, Istituto per la Matematica Applicata del C.N.R., Genova, Italy (1984).
5. J. Hagouel and M. Schwartz, Graphs partitioning with applications to computer networks, *Proceedings MELECON'83, Athens, 1983*, A5.04–5.
6. M. Harada and T. K. Kunii, A design process formalization, *Proceedings COMPSAC'79, Chicago 1979*, pp. 367–373.
7. F. Harary, *Graph Theory*, Addison-Wesley, Reading, Mass., 1977.
8. A. Iizawe, T. L. Kunii and S. Kawai, *A Graph-Based Hardware Design Specification System*, Technical Report 81–20, Department of Information Science, University of Tokyo, Japan (1981).
9. K. Kaerkes, Netzplan theory. *Methods of Operations Research*, **27**, 1–65 (1977).
10. C. A. Mead and L. A. Conway, *An Introduction to VLSI Systems*, Addison-Wesley, Reading, Mass. (1980).
11. R. H. Moehring and F. J. Radermacher, Substitution decomposition of discrete structures with applications to combinatorial optimization, *Annals of Discrete Mathematics* **19**, 257–356 (1984).
12. M. J. Quinn and N. Deo, *Parallel Algorithms and Data Structures in Graph Theory*, Research Report CS-82-098, Washington State University (1982).
13. C. V. Ramamoorthy and W. T. Tsai, An adaptive hierarchical routing algorithm, *Proceedings COMPSAC'83, Chicago, 1983*, pp. 93–99.
14. M. Silva, On the concept of macroplace and its use for Petri nets analysis, *RAIRO Automatique* **15** (4), 335–345 (1981), (in French).
15. A. S. Tanenbaum, *Computer Networks*, Prentice-Hall, Englewood Cliffs, N. J. (1980).
16. S. S. Yao and P. C. Grabow, A model for representing programs using hierarchical graphs, *IEEE Trans. on Software Engineering* **SE-7** (6), 556–573 (1981).