HARDWARE, SOFTWARE AND PEOPLEWARE

UNISA

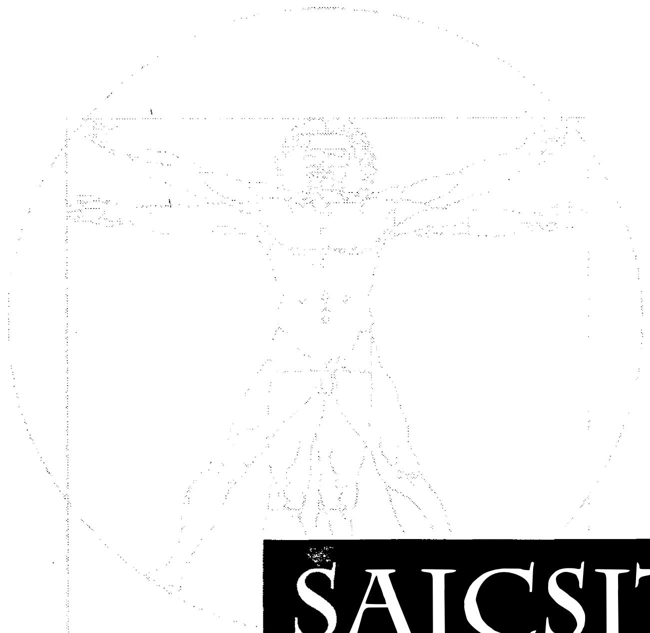
SAICSIT 2001

Edited by
Karen Renaud
Paula Kotzé
Andries Barnard

HARDWARE, SOFTWARE AND PEOPLEWARE

South African Institute of Computer Scientists and Information Technologists Annual Conference

25 – 28 September 2001
Pretoria, South Africa



SAICSIT 2001

Edited by Karen Renaud, Paula Kotzé & Andries Barnard
University of South Africa, Pretoria

Proceedings of the Annual Conference of the South African Institute of Computer Scientists and Information Technologists

First Edition, First Impression
ISBN: 1-86888-195-4

Published by Unisa Press
University of South Africa
P O Box 392, Pretoria, 0003

Cover Design by Tersia Parsons

Editors: Karen Renaud, Paula Kotzé & Andries Barnard

Electronic Publication by the Editors

Printed by Unisa Press
2001

Table of Contents

Keynote Speakers

Research Papers

Human-Computer Interaction / Virtual Reality

Education

Formal Methods

Human-Computer Interaction / Web Usability

Information Security

Abstracts of Electronic Papers

Message from the SAICSIT President

The South African Institute of Computer Scientists and Information Technologists (SAICSIT) was formed in 1982 and focuses on research and development in all fields of computing and information technology in South Africa. Now in the 20th year of its existence, SAICSIT has come of age, and through its flagship series of annual conferences provides a showcase of not only the best research from the Southern-African region, but also of international research, attracting contributions from far afield. SAICSIT does, however, not exist or operate in isolation.

More than 50 years have passed since the first electronic computer appeared in our society. In the intervening years technological development has been exponential. Over the last 20 years there has been a vast growth and pervasiveness of computing and information technology throughout the world. This has led into the expansion and consolidation of research into a diversity of new technologies and applications in diverse cultural environments. During this period huge strides have also been made in the development of computing devices. The processing speed of computers has increased thousand-fold and memory capacity from megabytes to gigabytes in the last decade alone. The Southern African region did not miss out on these developments.

It is hardly possible for such quantitative expansion not to bring a change in quality. Initially computers had been developed mainly for purposes such as automation for the improvement of processing, labour-reduction in production and automation control of machinery, with artificial intelligence, which made great strides in the 1980s, seen as the ultimate field to which computers could be applied. As we moved into the 1990s it was recognized that such an automation route was not the only direction in the improvement of computers. The expansion of processing power has enabled image data to be incorporated into computer systems, mainly for the purpose of improving human utilisation. For most computer technologies of the 1990s, including the Internet and virtual reality, automation was not the ultimate purpose. Humans were increasingly actively involved in the information-processing loop. This involvement has gradually increased as we move into the 21st century. Development of computer technology based not on automation, but on interaction, is now fully established.

The method of interaction has significantly changed as well. The expansion of computer ability means that the same function can be performed far more cheaply and on smaller computers than ever before. The advent of portable and mobile computers and pervasive computing devices is ample evidence of this. The need for users to be at the same location as a computer in order to reap the benefits of software installed on that computer is becoming an obsolete notion. Time and space are no longer constraints. One of the most discussed impacts of computing and information technology is *communication* and the easy accessibility of information. This changes the emphasis for research and development – issues such as cultural, political, and economic differences must, for example, be accommodated in ways that researchers have not previously considered. Our goal should be to enable users to benefit from technological advances, hence matching the skills, needs, and expectations of users of available technologies to their immense possibilities.

The conference theme for the SAICSIT 2001 Conference – *Hardware, Software and Peopleware: The Reality in the Real Millennium* – aims to reflect technological developments in all aspects related to computerised systems or computing devices, and especially reflect the fact that each influences the others.

Not only has SAICSIT come of age in the 21st century, but so has the research and development community in Southern Africa. The outstanding quality of papers submitted to SAICSIT 2001, of which only a small selection is published in this collection, illustrates both the exciting and developing nature of the field in our region. I hope that you will enjoy SAICSIT 2001 and that it will provide opportunities to cultivate and grow the seeds of discussion on innovative and new developments in computing and information technology.

Paula Kotzé
SAICSIT President

Message from the Chairs

Running this conference has been rewarding, exciting and exhausting. The response to the call for papers we sent out in March was overwhelming. We received 64 paper submissions for our main conference and twelve for the postgraduate symposium. We had a panel of internationally recognized reviewers, both local and international. The response from the reviewers was impressive – accepting a variety of papers and *mostly* returning the reviews long before the due date. We were struck, once again, by the sheer magnanimity of academia – as busy as we all are, we still manage to contribute fully to a conference such as SAICSIT.

After an exhaustive review process, where each paper was reviewed by at least three reviewers, the program committee accepted 26 full research papers and 14 electronic papers. Five papers were referred to the postgraduate symposium, since they represented work in progress – not yet ready for presentation to a full conference but which nevertheless represented sound and relevant research. The papers published in this volume therefore represent research of an internationally high standard and we are proud to publish it. Full electronic papers will be available on the conference web site (http://www.cs.unisa.ac.za/saicsit2001/).

Computer Science and Information Systems academics in South Africa labour under difficult circumstances. *The popularity of IT courses stems from the fact that IT qualifications are in high demand in industry, which leads in turn to a shortage of IT academic staff to teach the courses, even when posts are available. The net result is that fewer people teach more courses to more students. IT departments thus rake in ever-increasing amounts of state subsidy for their universities. These profits, euphemistically labelled "contribution to overhead costs", are deployed in various ways: cross-subsidization of non-profitable departments; maintenance of general facilities; salaries for administrative personnel, etc. Sweeteners of generous physical resources for the IT departments may be provided. We have yet to hear of a University in South Africa where significant concessions have been made in terms of industry-related remuneration. At best, small subventions are provided. As a result, shortages of quality staff remain acute in most IT departments – especially at senior teaching levels. What is even worse is that academics in these departments have to motivate the value of their conference contributions and other IT outputs to selection committees, often dominated by sceptical academic power-brokers from the more traditional departments whose continued survival is underwritten by IT's contribution to overhead costs.*[1]

The papers published in this volume are conclusive evidence of the indefatigability and pertinacity of Computer Science and Information Systems academics and technologists in South Africa. We are proud to be part of such a prestigious and innovative group of people.

In conclusion, we would like to thank the conference chair, Prof Paula Kotzé, for her support. We also specially thank Prof Derrick Kourie for his substantial contribution. Finally, to all of you, contributors, presenters, reviewers and organisers – a big thank you – without you this conference could not be successful.

Enjoy the Conference!
Karen Renaud & Andries Barnard

---

[1] This taken almost verbatim from Professor Derrick Kourie's SACLA 2001 paper titled: "*The Benefits of Bad Teaching*".

Conference Organisation

General Chair
Paula Kotzé

Programme Chairs
Karen Renaud
Andries Barnard

Organising Committee Chairs
Lucas Venter, Alta van der Merwe

Art and Design
Tersia Parsons

Sponsor Liaison
Paula Kotzé, Chris Bornman

Secretarial & Finances
Christa Prinsloo, Elmarie Havenga

Marketing & Public Relations
Klarissa Engelbrecht, Elmarie van Solms, Adriaan Pottas, Mac van der Merwe

Audio Visual
Tobie van Dyk, Andre van der Poll, Mac van der Merwe

Program Committee

Bob Baber – McMaster Univeristy, Canada
Andries Barnard – University of South Africa
Judy Bishop – University of Pretoria
Andy Bytheway – University of the Western Cape
Andre Calitz – University of Port Elizabeth
Elsabe Cloete – University of South Africa
Carina de Villiers – University of Pretoria
Alan Dix – Lancaster University, United Kingdom
Jan Eloff – Rand Afrikaans University
Andries Engelbrecht – University of Pretoria
Chris Johnson – University of Glasgow, United Kingdom
Paul Licker – University of Cape Town
Paula Kotzé – University of South Africa
Derrick Kourie – University of Pretoria
Philip Machanick – University of the Witwatersrand
Gary Marsden – University of Cape Town
Don Petkov – University of Natal in Pietermaritsburg
Karen Renaud – University of South Africa
Ian Sanders – University of the Witwatersrand
Derrick Smith – University of Cape Town
Harold Thimbleby – Middlesex University, United Kingdom
Theda Thomas – Port Elizabeth Technikon
Herna Viktor – University of Pretoria, South Africa
Bruce Watson – Universities of Pretoria and Eindhoven
Janet Wesson – University of Port Elizabeth

Referees

Molla Alemayehu
Trish Alexander
Adi Attar
Bob Baber
Andries Barnard
John Barrow
Judy Bishop
Gordon Blair
Arina Britz
Andy Bytheway
André Calitz
Charmain Cilliers
Elsabe Cloete
Gordon Cooper
Richard Cooper
Annemieke Craig
Thad Crews
Quintin Cutts
Michael Dales
Carina de Villiers
Alan Dix
Dunlop Mark
Elize Ehlers
Jan Eloff
Andries Engelbrecht
Klarissa Engelbrecht
David Forsyth
John Galletly
Vashti Galpin
Wayne Goddard
Alexandré Hardy
Scott Hazelhurst
Johannes Heidema
Tersia Hörne
Chris Johnson
Bob Jolliffe
Paula Kotzé
Derrick Kourie
Les Labuschagne
Paul Licker
Philip Machanick
Anthony Maeder
David Manlove
Gary Marsden
Thomas Meyer
Elsa Naudé
Martin Olivier
Don Petkov
Pekka Pihlajasaari
Nelisha Pillay
Laurette Pretorius
Karen Renaud
Ingrid Rewitzky
Sheila Rock
Markus Roggenbach
Ian Sanders
Justin Schoeman
Martie Schoeman
Elsje Scott
Derek Smith
Elmé Smith
Adrie Stander
Harold Thimbleby
Theda Thomas
Judy Van Biljon
Alta Van der Merwe
André van der Poll
Tobias Van Dyk
Lynette van Zijl
Lucas Venter
Herna Viktor
Bruce Watson
Janet Wesson

Conference

Sponsors

UNISA
ABSA Group Groep
Microsoft
ORACLE
Libra
NRF

Keynote Abstracts

Combining context provisions with graph grammar rewriting rules - the three-dimensional case

A Barnard [a] E M Ehlers [b]

[a] *University of South Africa, South Africa* barnaa@unisa.ac.za

[b] *Rand Afrikaans University, South Africa,* eme@na.rau.ac.za

Abstract: *In this paper we consider the representation and generation of three-dimensional structures by means of formal descriptive methods. Graph and graph grammar theory present us with a powerful two-dimensional representational method, and we propose to use these concepts as basis for the three-dimensional case. Three-dimensional structures however, often appear in other structures and within a certain context. This context may be defined or influenced by the overall structure, or other related structures. We therefore need to be cognisant of the role that these contexts play when we introduce the concepts of three-dimensional graph and graph grammar systems, with particular reference to contextual rewriting rules. It is the combination of context provisions with graph grammar rewriting rules that results in a formal descriptive method which represents three-dimensional structures. The generative abilities of the concepts we introduce and discuss, are illustrated by considering the generation of various chemical structural formulae.*

Keywords: *Formal language, grammar theory, graph grammars, context sensitive rewriting rules*

Computing Review Categories: F.4.2, F.4.3

1. Introduction

In this paper we consider the representation and generation of three-dimensional structures by means of formal descriptive methods. Graph and graph grammar theory present us with a powerful two-dimensional representational method [1, 6, 11, 12, 14], and we propose to use these concepts as basis for the three-dimensional case. In section 2 we therefore review some of the basic notions regarding graph and graph grammar theory. Three-dimensional structures often appear in other structures and within a certain context. It is often difficult to describe these contextual relations succinctly and in sections 3 and 4 we show how these contexts can be combined with graphs and, in particular, graph grammar rewriting rules. These contextualised graph grammar rewriting rules will enable us to describe and generate three-dimensional structures, and we illustrate this statement by considering the generation of various chemical structural formulae. We conclude by stating the obvious: namely that the descriptive abilities of these new grammars are clearly useful.

2. Graphs and graph grammars

2.1 Definition of a graph:

We give the definition of a graph as in [6]:

> Let X be a finite alphabet. A *graph* G = (V, E, m) is a simple, undirected node labelled graph and consists of a finite set of vertices or nodes V, a set of undirected edges E without self-loops and multiple edges, and a vertex labelling function m: $V \rightarrow X$. G is then indicated by (V(G), E(G), m(G)). □

The theory of graphs also allows for a directed analogue, i.e. where every edge of the graph is assigned a specific direction. Such a directed graph is referred to as a digraph for short. For more information on graph theory and related topics, the reader is referred to [7] that contains an exhaustive and recent bibliography.

2.2 Graph grammars

Picture recognition and picture processing are areas of research receiving much attention. Different strategies have been followed to address this problem, see for example [2, 9, 10, 13, and 17]. One of the problem areas identified is to transfer the basic properties of string grammars to general structures. Pfaltz and Rosenfeld defined graph grammars in 1969, see [15], where replacement of a symbol can occur in any direction. The restriction of string grammars where replacement of a symbol can occur only left or right of the symbol in a string, was thus something of the past. We now review graph grammars from the family of vertex replacement systems as in [6]:

> A *Graph Grammar* is a four-tuple GG = (N,T,S,P):
> - N is the alphabet of nonterminal vertex labels,
> - T is the alphabet of terminal vertex labels,
> - S is the start axiom where $S \in N - T$, and
> - P is a finite set of productions.
>
> Every production from P is of the form p=(A,R,C):
> - A is the vertex label of the left-hand side of the production and indicates the vertex to be replaced, where $A \in N$,
> - R is a nonempty graph to replace A and
> - C is the connection relation consisting of

pairs (a, w) with $a \in (N \cup T)$ and $w \in V(R)$. □

Notational conventions:

- According to the vertices' labels, we refer to vertices as being terminal and nonterminal.
- For a vertex $w \in V(R)$, let $C^{-1}(w) = \{a | a \in (N \cup T), (a, w) \in C\}$ denote the set of labels in the connection relation of w.
- We will throughout the rest of this section make use of the following graphic notation for the production rules P of GG:
  For p = (A, R, C) of P, draw the right-hand side graph R with the nonterminal vertices as unit size squares, terminal vertices as points and (whenever possible) straight line edges. For the left-hand side, draw a big rectangle with label A around R. Finally, for every connection $(a, w) \in C$, draw a line from the vertex w of R to an a-labelled point outside the big rectangle.
- A derivation step of GG where (A, R, C) is a production of P, means replacing a vertex v with label A by the right-hand side R and establishing connections between the neighbours of v and the vertices of R as specified by C.
- The language $\mathcal{L}$(GG) generated by GG, consists of all graphs with only terminal vertices that can be derived from the axiom S.

For more information on this, see [6].

An example:

In this subsection, we give an example of a graph grammar that generates chains of the form $a^n b^n c^n$ where $n \geq 1$.

Let GG = (N, T, S, P) where:
N = {S, A},
T = {a, b, c} and
P = {P1, P2, P3, P4} where the productions are shown graphically in Figures 1 through 4:

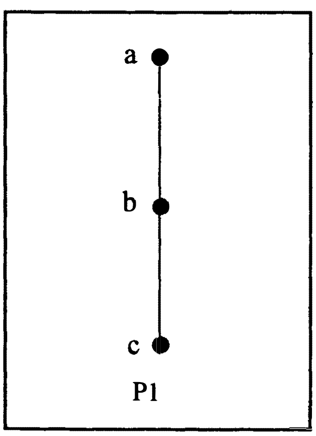


Figure 1 - The first production of GG

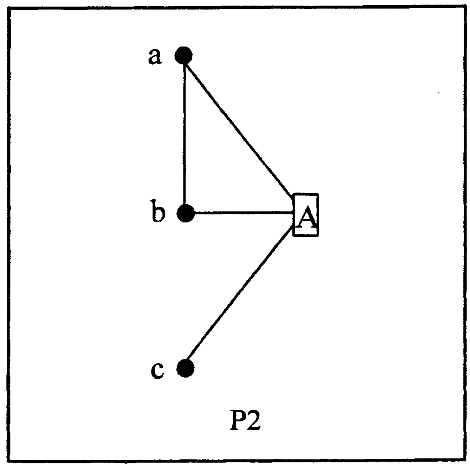


Figure 2 - The second production of GG

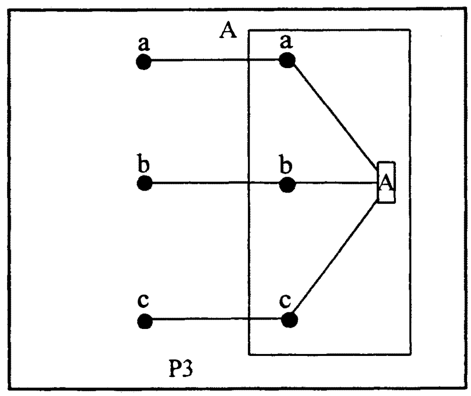


Figure 3 - The third production of GG

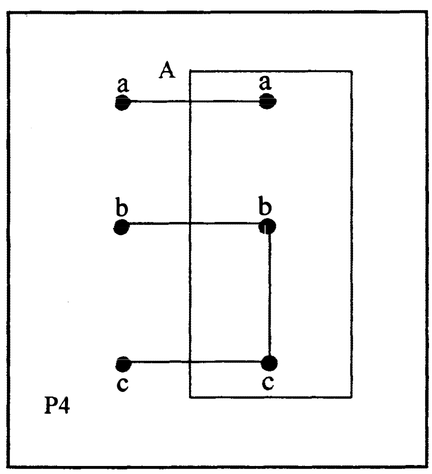


Figure 4 - The fourth production of GG

A possible derivation of the above grammar is illustrated graphically below, Figures 5 through 8:

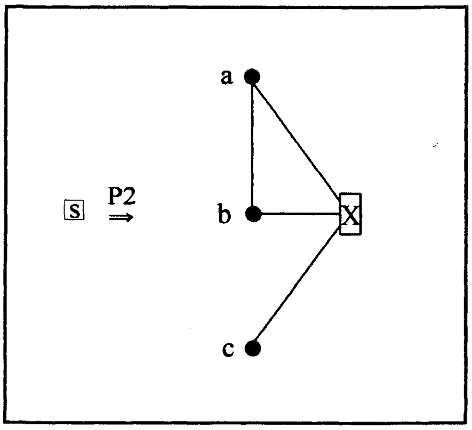


Figure 5 - Application of P2

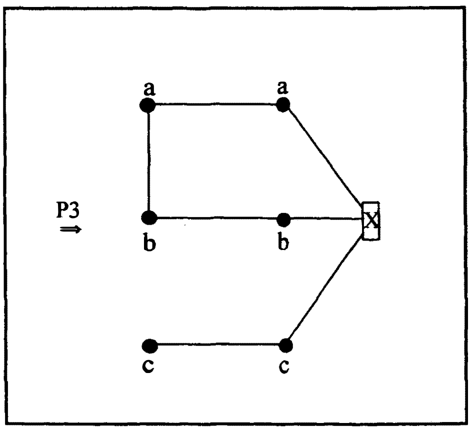


Figure 6 - Application of P3

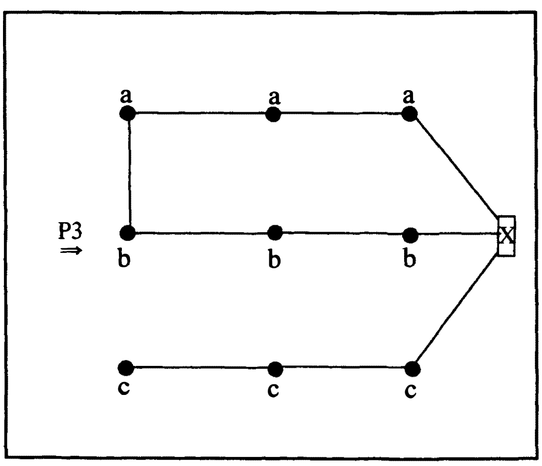


Figure 7 - Application of P3

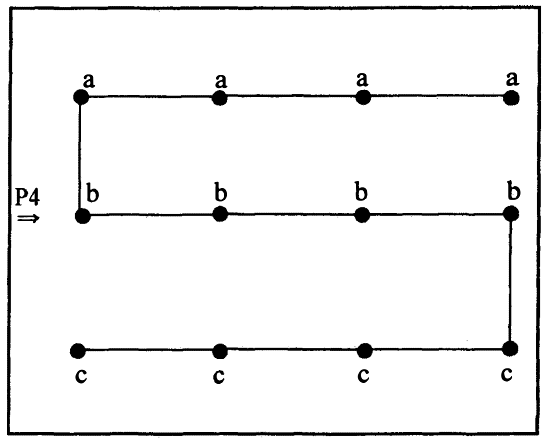


Figure 8 - Application of P4

The language generated by this graph grammar GG is all chains of the form $a^n b^n c^n$ where $n \geq 1$.

There are many classes of graph grammars to be found in the literature. Pratt considered the generation of digraphs in [16], whereas Montanari introduced so-called web grammars in [14], also see [1]. Janssens and Rozenberg suggested in [11, 12], that it is not yet clear which concepts in graph grammar theory are of importance, and which concepts are of lesser importance. For example, mathematical manipulations to succinctly describe graph grammar productions as well as graph languages, are as yet unsatisfactory. In order to establish a concise description of graph grammar languages, they defined NLC-grammars (Node-Label-Controlled Graph Grammars) and introduced a mathematical manipulation rule in order to describe these NLC-graph languages, [11, 12].

3. The three-dimensional case: Structure Graphs

In the introduction we mentioned that the contexts within which a three-dimensional structure appear, is often influenced by other structures. In order to make provision for the accurate description of these contexts, we need to be able to describe them formally. Few three-dimensional generative devices allow the succinct description of these contexts. As graph grammars present a powerful two-dimensional generative device, we propose to combine graph rewriting rules with context provisions in the following section to accomplish this.

3.1 Structure graphs

In order to generate three-dimensional structures with the use of graph grammars, we first of all need to be able to describe three-dimensional structures in a graph theoretic manner with particular reference to accurate description of contextual constraints that we are interested in. In [3, 4] we gave the following definition of a structure parameter:

A *structure parameter* p is a real-valued number that either gives a geometrical property of an edge of a graph (e.g. length of the edge) or describes the relationships of two edges in a graph (e.g. angle between two edges in the graph). □

We now give the definition of a structure graph, [3]:

A *structure graph* (SG) G is a 2-tuple (V, E) where V is the vertex set and E the structured edge set of G. An element of E has general form:

$((v_iv_j, p_{ij_1}, p_{ij_2}, \ldots, p_{ij_m}),\ (v_kv_l, p_{kl_1}, p_{kl_2}, \ldots, p_{kl_m}),\ p_1, p_2, \ldots, p_r)$

where $v_i, v_j, v_k, v_l \in V$ ($i, j, k, l \in \{1, 2, 3, \ldots, |V|\}$) and $p_{ij_1}, \ldots, p_{ij_m}, p_{kl_1}, \ldots, p_{kl_m}, p_1, \ldots, p_r$ are structure parameters ($m, r \in \mathbb{N}$). □

The value and meaning of a structure parameter will be assigned by convention as needed with consideration of the specific structure graph in question. We remark that if for a structure parameter p_i, $ij_1 \le i \le ij_m$, $kl_1 \le i \le kl_m$ or $1 \le i \le r$, we have $p_i = 0$, p_i is removed from the definition of a structure graph. We further say that the *underlying graph* of a structure graph is merely the graph for which all structure parameters' values are negated (thus the graph without geometrical properties). The underlying graph of a structure graph S will be written as *und (S)*.

If *x* is a graph theoretical property, we say the structure graph S has graph property *x* iff *und (S)* has graph property *x*. □

In the rest of this paper we limit our attention to the following instance of a structure graph, namely an *angle-length structure graph:*

An *angle-length structure graph* (AL-SG) is a structure graph G where $p_i = 0$ for all $i \in \{ij_2, \ldots, ij_m, kl_2, \ldots, kl_m, 2, \ldots, r\}$. The structure parameters p_{ij_1}, p_{kl_1}, and p_1 are then assigned the following geometrical properties:

- p_{ij_1} gives the length of the edge v_iv_j,
- p_{kl_1} gives the length of the edge v_kv_l, and
- p_1 gives the angle constituted by the edges v_iv_j and v_kv_l in G. □

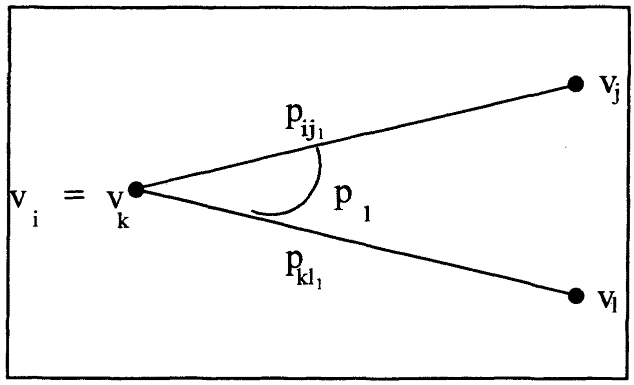


Figure 9 - Graphical interpretation of AL-SG ($v_i = v_k$)

3.2 Examples of AL-SG:

Below we give some examples of how this definition can be used to describe three-dimensional objects, with particular reference to the contexts that we wish to represent. First of all, consider the example of a unit cube:

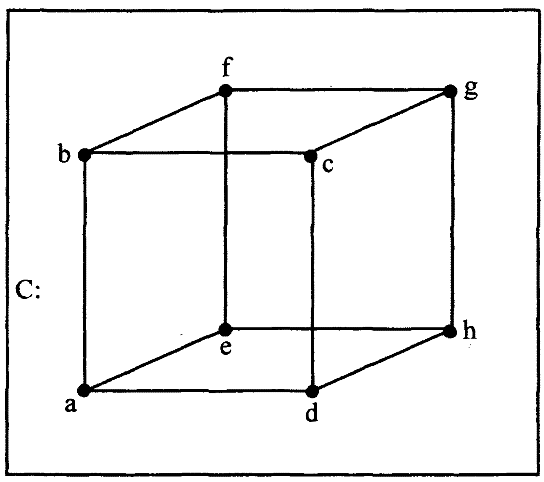


Figure 10 - a cube C

The formal description of C in terms of an angle-length structure graph is as follows:

C = (V, E) where:

V = {a, b, c, d, e, f, g, h} and

E = {((ab,1), (bc,1), 90°); ((bc,1), (cd,1), 90°); ((cd,1), (da,1), 90°); ((da,1), (ab,1), 90°); ((bc,1), (ad,1), 0°); ((ab,1), (dc,1), 0°); ((bc,1), ((bf,1), 90°); ((ab,1), (bf,1), 90°); ((bf,1), (ef,1), 90°); ((bf,1), (fg,1), 90°); ((fg,1), (cg,1), 90°); ((bf,1), (cg,1), 0°); ((cd,1), (cg,1) 90°); ((bc,1), (cg,1), 90°); ((cg,1), (gh,1), 90°); ((gh,1), (hd,1), 90°);

((cd,1), (dh,1), 90°); ((ae,1), (dh,1), 0°); ((ae,1), (eh,1), 90°); ((ad,1), (ae,1), 90°); ((ad,1), (eh,1), 0°)}.

We now turn our attention to the representation of chemical structural formulae by structure graphs. In all cases information pertaining to structural formulae of chemical compounds referenced here, as well as scan images, are from [5]. As point of departure, we consider the first member of the homologous chain of alcanes and the basis of a vast amount of organic compounds, namely methane with chemical formula CH_4, i.e. one carbon atom bonded to four hydrogen atoms to form a tetrahedron in space. The planar representation of the structural formula of a methane molecule is given in figure 11 below:

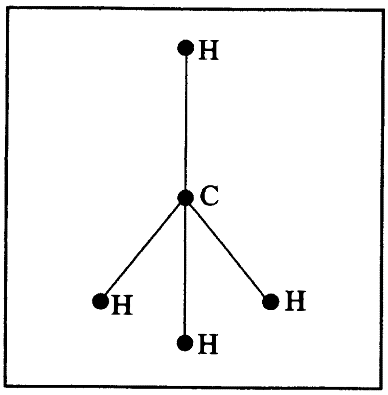


Figure 11 - Methane (CH_4)

In the structural formulae of chemical molecules, bond lengths as well as bond angles constitute the contexts within which the atoms appear, and are of importance. In this respect the Carbon - Hydrogen bond length is 1,10Å while all C-H-C bond angles are 109,5°. Figure 12 is a scan image of the three-dimensional conformation of a methane molecule.

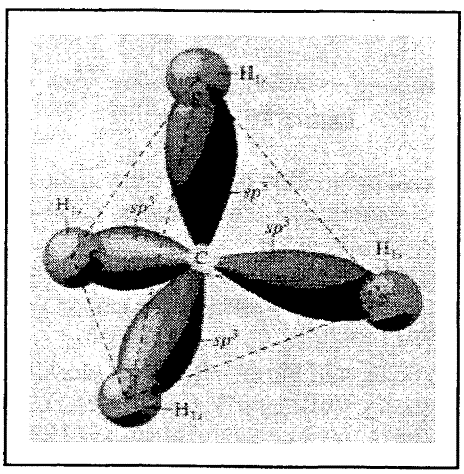


Figure 12 - scan image of a methane molecule

The following definition of an angle-length structure graph gives a complete and formal description of a methane molecule:

M = (V, E) where:

V = $\{C, H_1, H_2, H_3, H_4\}$ and

E = $\{((C - H_1, 1{,}10Å),(C - H_2, 1{,}10Å), 109{,}5°);$
$((C - H_2, 1{,}10Å),(C - H_3, 1{,}10Å), 109{,}5°);$
$((C - H_1, 1{,}10Å),(C - H_3, 1{,}10Å), 109{,}5°);$
$((C - H_1, 1{,}10Å),(C - H_4, 1{,}10Å), 109{,}5°);$
$((C - H_3, 1{,}10Å),(C - H_4, 1{,}10Å), 109{,}5°);$
$((C - H_2, 1{,}10Å),(C - H_4, 1{,}10Å), 109{,}5°)\}.$

The following example is that of an angle-length structure graph describing a propane molecule. Propane is the third member in the homologous chain of alcanes and the planar representation of propane is given in figure 13, whereas figure 14 gives a scan image of a propane molecule. The carbon to carbon bond lengths are all 1,53Å while the carbon to hydrogen bond lengths are all 1,10Å. The C-C-C, H-C-H, C-H-H, H-H-C and C-C-H bond angles are all 109,5°.

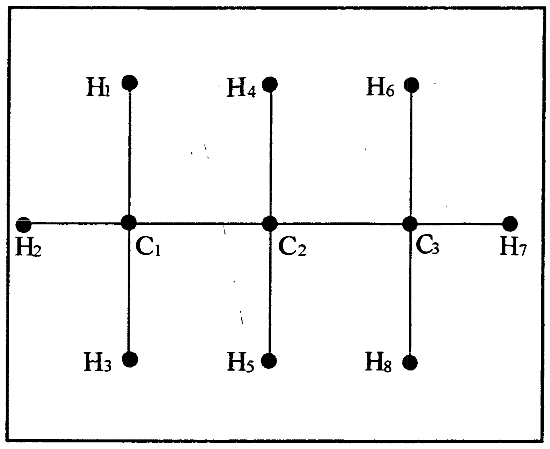


Figure 13 - Propane (C_3H_8)

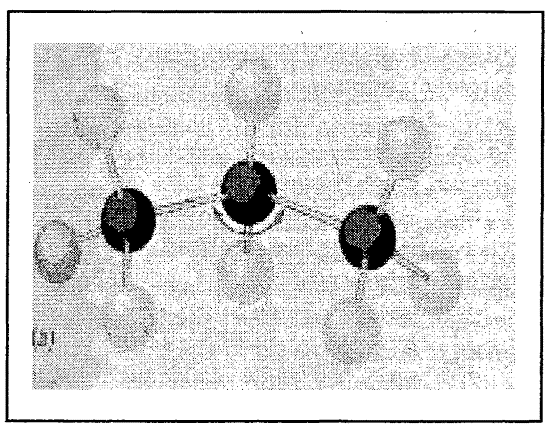


Figure 14 - scan image of a propane molecule

The following definition of an angle-length structure graph gives a complete and formal description of a propane molecule:

P = (V, E) where:

V = $\{C_1, C_2, C_3, H_1, H_2, H_3, H_4, H_5, H_6, H_7, H_8\}$;

E = $\{((C_1 - H_1, 1{,}10Å),(C_1 - H_2, 1{,}10Å), 109{,}5°)$;
$((C_1 - H_2, 1{,}10Å),(C_1 - H_3, 1{,}10Å), 109{,}5°)$;
$((C_1 - H_1, 1{,}10Å),(C_1 - H_3, 1{,}10Å), 109{,}5°)$;
$((C_1 - H_1, 1{,}10Å),(C_1 - C_2, 1{,}53Å), 109{,}5°)$;
$((C_1 - H_2, 1{,}10Å),(C_1 - C_2, 1{,}53Å), 109{,}5°)$;
$((C_1 - H_3, 1{,}10Å),(C_1 - C_2, 1{,}53Å), 109{,}5°)$;
$((C_2 - H_4, 1{,}10Å),(C_2 - H_5, 1{,}10Å), 109{,}5°)$;
$((C_2 - H_4, 1{,}10Å),(C_1 - C_2, 1{,}53Å), 109{,}5°)$;
$((C_2 - H_5, 1{,}10Å),(C_1 - C_2, 1{,}53Å), 109{,}5°)$;
$((C_2 - H_4, 1{,}10Å),(C_2 - C_3, 1{,}53Å), 109{,}5°)$;
$((C_2 - H_5, 1{,}10Å),(C_2 - C_3, 1{,}53Å), 109{,}5°)$;
$((C_1 - C_2, 1{,}53Å),(C_2 - C_3, 1{,}53Å), 109{,}5°)$;
$((C_3 - H_6, 1{,}10Å),(C_3 - H_7, 1{,}10Å), 109{,}5°)$;
$((C_3 - H_7, 1{,}10Å),(C_3 - H_8, 1{,}10Å), 109{,}5°)$;
$((C_3 - H_6, 1{,}10Å),(C_3 - H_8, 1{,}10Å), 109{,}5°)$;
$((C_3 - H_6, 1{,}10Å),(C_2 - C_3, 1{,}53Å), 109{,}5°)$;
$((C_3 - H_7, 1{,}10Å),(C_2 - C_3, 1{,}53Å), 109{,}5°)$;
$((C_3 - H_8, 1{,}10Å),(C_2 - C_3, 1{,}53Å), 109{,}5°)\}$.

If we omit the subscripts we obtain a shorter and more elegant structured edge set that still describes the same molecule (non-deterministically in this case as the number of carbon atoms can be decided on randomly):

E′ = {((C - C, 1,53Å), C - C, 1,53Å), 109,5°);
((C - C, 1,53Å), C - H, 1,10Å), 109,5°);
((C - H, 1,10Å), C - H, 1,10Å), 109,5°).

4. The generation of structure graphs

We now turn our attention to a class of graph grammars that can generate these structure graphs. As we have seen above, inclusion of the contexts in the definition of a structure graph, enables us to describe the three-dimensional attributes of structures succinctly. We therefore need to include such contexts in the definition of the class of grammars that must generate structure graphs.

4.1 Structure Graph Grammars (SGG):

In order to define a structure graph grammar, we need to formalise what we understand under the term *conflict:*

> By the term *conflict* we will mean the competition of two vertices (either with the same or with different labels) to occupy the same physical space, e.g. spatial position, during the application of a production rule to a structure graph. □

The definition of a structure graph in this paper, is similar to that of SGG-4 as in [3] with the exception that we will only consider the definition relative to angle-length structure graphs (AL-SG) in this paper:

A *structure graph grammar* (SGG) is defined as a 4-tuple G = (N, T, S, R) where:
- N is a set of nonterminal symbols,
- T is a set of terminal symbols,
- S is the start symbol (a single vertex), and
- R is the set of production rules.

A production rule r ∈ R is defined as a 3-tuple r = (F, L, C) where:
- F is a vertex or an edge in the structure graph to be replaced,
- L is a vertex or an edge to replace F, and
- C is the context.

For the case of an AL-SG and F an edge, F has general form F = (F, *f*) where:
- F is the edge in the AL-SG and
- *f* is the length of edge F.

The same will apply if L is an edge.

The context C is defined as a 3-tuple $C=(C_h / U, T_1)$ where:
- C_h is the angle context,
- U is the global permitting context, and
- T_1 is the global forbidding context ($U, T_1 \subseteq N \cup T$).

An element of the angle context has general form (L_1, a) where L_1 is an edge in the AL-SG under consideration and *a* is the angle between edge L (in the production rule r) and the edge L_1 (in the AL-SG). Note that if both F and L are vertices or F is an edge and L a vertex, the angle context is superfluous and will be omitted.

Consider the rule r = (F, L, C) to be applied to an AL-SG, say S. The production rule r can be applied to S if and only if F is present in S and context C holds. Context C holds if and only if:
(1) All the elements of U are vertices or edges in S.
(2) None of the elements of T_1 are vertices nor edges in S.
(3) All edges as specified in the angle-context C_h, must be present in S.
(4) F must be replaced with L such that the angle(s) as specified in C_h, will hold.

In order to define the replacement operation of F with L, we consider the following four instances:
(1) Both **F** and **L** are **vertices.** This entails a label-replacement of F with L.
(2) **F** a **vertex** and **L** an **edge**, say $L = n_1 - n_2$. Note that F can be replaced with L if and only if one of the vertices of L, is F, thus

either $n_1 = F$ or $n_2 = F$. Then:

(2.1) Only one of n_i ($i = 1,2$) is present in S (it has to be $n_i = F$, $i = 1, 2$). We replace vertex F with edge L such that all contexts hold. Because we assume that only *one* of the vertices of the edge (whatever the vertex labels may be) is present in the structure graph, no conflict can arise in this case.

(2.2) Both vertices n_1 and n_2 are present in S where say $F = n_1$. To replace F with L, we distinguish the following two instances:

(i) Context C specifies that vertex n_2 of edge L has to be placed in S such that the spatial position of this vertex n_2 and the spatial position of vertex n_2 already present in S, are different. Vertex F ($= n_1$) is then replaced with edge $L = (n_1 - n_2)$ such that all contexts hold. No conflict can arise in this case.

(ii) Context C specifies that vertex n_2 of edge L has to be placed in S such that the spatial position of this vertex n_2 and the spatial position of vertex n_2 already present in S, are the same. We then say that the two vertices n_2 are to be identified and proceed as follow in order to replace vertex $F(= n_1)$ with edge $L (= n_1 - n_2)$:

Add a new vertex to S, specifically an *invisible vertex*, denoted by tg and such that contexts where n_2 is replaced with tg in the applicative angle-context-set, will hold. Connect tg with all vertices in S that are adjacent to vertex n_2 in S. Remove n_2 and all incident edges from the structure graph. Vertex F ($= n_1$) can be replaced with edge $L (= n_1 - n_2)$ as in case 2.2.(i) such that all contexts hold where vertex n_2 is identified with vertex tg. Note that n_2 is now present in the structure graph with spatial position of n_2 the same as the spatial position of tg because of the context-specifications. Connect vertex n_2 with a vertex x in the structure graph if and only if tg and x are adjacent. Remove tg and all incident edges from the structure graph in order to obtain the resulting structure graph. This case makes provision for conflict between two vertices with the same labels.

Case 2.2 is the same if $F = n_2$. We then replace n_1 with n_2 and vice versa throughout case 2.2.

(3) **F** is an **edge**, say $F = n_1 - n_2$, and **L** is a **vertex**. Note that F can be replaced with L if and only if L is one of the end-vertices of F, thus $L = n_1$ or $L = n_2$. Assume $L = n_2$. We replace F with L as follows:

Remove n_1 as well as all incident edges from the structure graph (note that for $n_1 \neq L$, we have $n_2 = L$ still present in the structure graph). No conflict can arise in this case.

(4) Both **F** and **L** are **edges**. Note that edge F may be replaced with edge L if and only if the edges F and L have a vertex in common, referred to as the *grip-vertex* of the edges F and L. Say $F = n - m_1$ and $L = n - m_2$, thus n is the grip-vertex of F and L. In order to replace F with L, we proceed as follow:

Remove the vertex of F that is not the grip-vertex of F and L as well as all incident edges from the structure graph. Note that the grip-vertex is still present in the structure graph. Replace the grip-vertex with edge L as in case 2 such that all contexts hold.

If the application of any of the production rules to a structure graph results in the occurrence of conflict, the generative process is aborted and no structure graph generated. □

4.2 Examples of SGG application:

For illustrative purposes of production rule application to a structure graph according to the above definition, consider the following scenario where S is the structure graph in the plane given in figure 15:

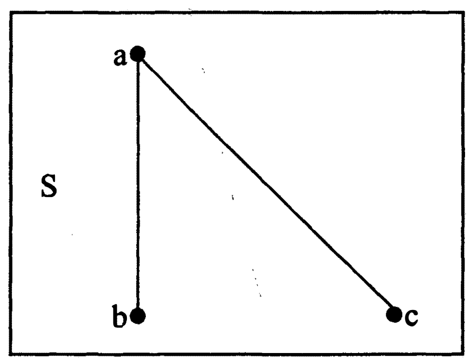


Figure 15 - The structure graph S

Let the production rule r to be applied to S be:

r = (b, b - c, {(a - b, 90°); (a - c, 45°)} / {a}, { }).

This means that vertex b is to be replaced by edge b - c. Note however that both vertices b and c are already present in the structure graph and the context specifications, specifically the angle-contexts, furthermore specify that vertex c of edge b - c must occupy the same spatial position than that of vertex c already present in S. This is due to the fact that the angle formed between edges a - b and b - c must be 90° and the angle formed between edges a - c and b - c must be 45°. We thus follow case 2.2(ii) of the above definition and make use of the concept of an invisible vertex. The reason for this is perhaps not entirely intuitive at first glace as it seems straightforward that one should simply connect the vertices b and c. We have to bear in mind however that grammar systems only rely on the rewriting rules and the way in which these rewriting rules are defined. There are no outside influence to suggest that the vertices b and c should be connected. According to the graph rewriting rule principles we have to replace a subgraph (which in this case is the vertex b) with another subgraph (in this case the edge b-c). However the contexts now specify that the vertex c of this subgraph (of the edge b-c to replace vertex b) and the vertex c that is already present in the subgraph, must occupy the same spatial position. This is akin to specifying that 2 different entities must occupy the same physical position. This is thus where the concept of an *invisible vertex* proves to be useful. We therefore apply case 2.2(ii) of the definition to this example. First of all vertex c is replaced by an invisible vertex:

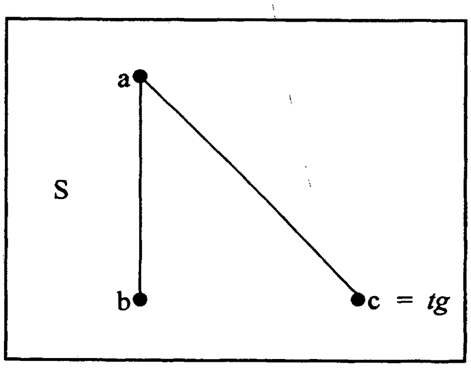


Figure 16 - Vertex c replaced by *tg*

We can now replace vertex b with edge b-c where vertex c and the invisible vertex are identified in the context set.

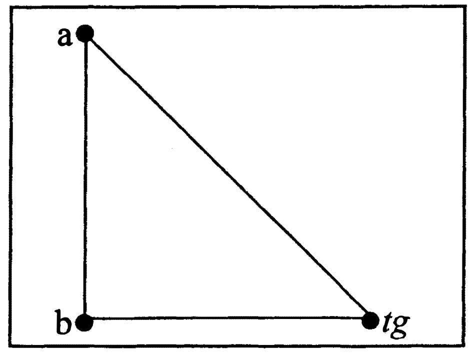


Figure 17 - Replacement of vertex b by edge

Vertex c and vertex tg are now occupying the same spatial position. We thus proceed by removing vertex tg from S.

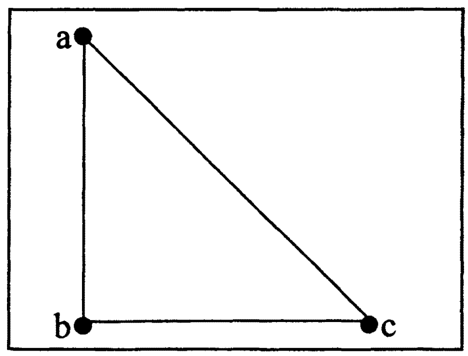


Figure 18 - The final cyclic structure graph

The following example of a SGG can generate the homologous chain of alcanes (see [5] for more information concerning the family of alcanes, as well as general chemistry), G = (N,T,S,R) where:

N = {s_C, s_H},
T = {C, H},
S = {s_C}, and
R = {

(1) (s_C, (s_C - s_H, 1,10Å), { } / {s_C}; {s_H});
(2) (s_C, (s_C - s_H, 1,10Å), {((s_C - s_H, 1,10Å), 109,5°)} / {s_C, s_H}, {C, H});
(3) (s_C, (s_C - s_H, 1,10Å), {((s_C - s_H, 1,10Å), 109,5°), ((s_C - s_C, 1,53Å), 109,5°)} / {s_C, s_H}, {C, H});
(4) ((s_C - s_H, 1,10Å), (s_C - s_C, 1,53Å), {((s_C - s_H, 1,10Å}, 109,5°), ((s_C - s_C, 1,53Å), 109,5°)} / {s_C, s_H}, {C, H});
(5) (s_C, C, { } / {s_C, s_H}, {H});
(6) (s_H, H, { } / {s_H, C}, {s_C})}.

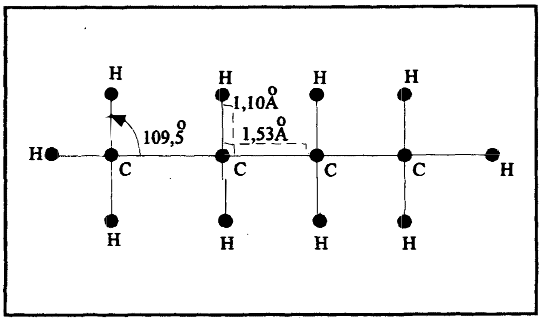


Figure 19 - An alcane (Butane C_4H_{10})

It is the ability of SGG to replace an edge with another (of possibly different length) as illustrated in production rule no. (4) in the above grammar, that enables SGG to generate the chain of alcanes as an edge s_C - s_H of length 1,10Å can be replaced with the edge s_C - s_C of length 1,53Å and eventually the functional group $s_C\, s_{H3}$ (equivalently CH_3, a methyl group).

The following SGG will yield a benzene molecule, G = (N, T, S, R) where:

N = {s_1, s_2, s_3, s_4, s_5, s_6},
T = {C_1, C_2, C_3, C_4, C_5, C_6, H},
S = {s_1}, and
R = {

(1) (s_1, (s_1 - H, 1,10Å), (/));
(2) (s_1, (s_1 - s_2, 1,39Å), ({((s_1 - H, 1,10Å), 120°)} / { }; {s_2}));
(3) (s_i, (s_i - s_{i+1}, 1,39Å), ({((s_i - H, 1,10Å), 120°), ((s_i - s_{i-1}, 1,39Å), 120°)} / { }, { })) $2 \le i \le 5$;
(4) (s_6, (s_6 - s_1, 1,39Å), ({((s_6 - H, 1,10Å), 120°); ((s_1 - s_2, 1,39Å), 120°), ((s_5 - s_6, 1,39Å), 120°)} / { }, { }));
(5) (s_i, (s_i - H, 1,10Å), ({((s_i - s_{i-1}, 1,39Å), 120°); ((s_j - s_{j+1}, 1,39Å), 120°)} / {s_1, s_2, s_3, s_4, s_5, s_6}, {C_1, C_2, C_3, C_4, C_5, C_6})) $2 \le i \le 6$, $1 \le j \le 5$;
(6) (s_i, C_i (/ {s_i}, { })) $1 \le i \le 6$}.

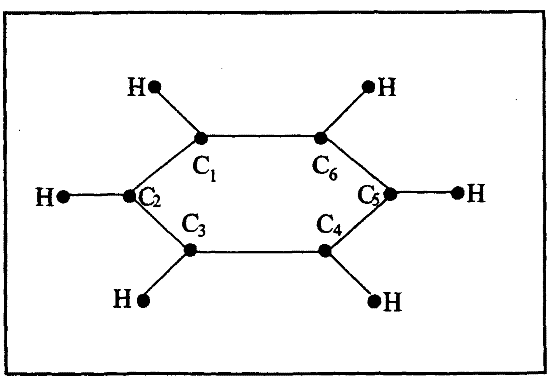


Figure 20 - Benzene

From the examples above it is clear that SGG can generate cyclic (figure 20) as well as acyclic (figure 19) structures. It is also clear that SGG can generate connected structures. Furthermore, if we apply rewriting rule 3 (edge with vertex replacement) to a bridge in a connected SG (a bridge is an edge uv of a graph G such that G-uv is disconnected), the resultant structure will be disconnected. We thus have the following corollary:

SGG can generate cyclic, acyclic, connected as well as disconnected structures. □

5. Conclusion

In this paper we considered the representation of three-dimensional structures by means of graphs, and we presented a way in which these structures can be described as graphs such that the context provisions of the structures, are present in the definition. We then considered the combination of these context provisions with graph grammar rewriting rules and showed how these rules could be used to generate three-dimensional structures, in particular the structural conformation of compounds found in organic chemistry. In [18] Van der Walt and Ewert considered the influence of random use of different types of contexts in grammar rewriting rules, particularly forbidding and permitting context sets. They furthermore refer to its possible application in syntactic picture generation.

We thus conclude with the contention that our contribution to the study of three-dimensional generative devices, and in particular the combination of context provisions with general graph rewriting rules, may yield positive results compared to more primitive devices that generate digital structures [8, 19, 20].

6. References

[1] Abe, N., Mizumoto, M., Toyoda, J. and Tanaka, K.; *"Web Grammars and Several Graphs"*; Journal of Computer and System Sciences; **Vol. 7** (1973).

[2] Ahuja, N. and Veenstra, J.; *"Line drawings of Octree-Represented Objects"*; A.C.M. Transactions on Graphics; **Vol. 7**, No. 1 (1988).

[3] Barnard, A; *Structure graph grammars and structure graph automata*; PhD thesis; RAU; Johannesburg (1997).

[4] Barnard, A. and Ehlers, E. M.; *"A comparative study of random context structure grammars and structure graph grammars"*; Combinatorics, Graph theory, and Algorithms;

New Issues Press; Western Michigan University; Kalamazoo; MI (1999).

[5] Boyd, R.W. and Morrison, R.T.; *Organic Chemistry, 5th Edition*; Allyn and Bacon Inc.; Boston, MA (1987).

[6] Brandenburg, F.J. and Skodinis, K.; *"Graph Automata for Linear graph Languages"*; Graph Grammars and their Application to Computer Science; Cuny et al. (Eds.); Springer-Verlag (1995).

[7] Chartrand, G. and Lesniak, L.; *Graphs & Digraphs, Third edition*; Chapman & Hall, London (1996).

[8] Ehlers, E. M. and Von Solms, S.H.; *"Random Context Structure Grammars and Automata - A Formal Approach"*; Information Sciences; **53** (1991).

[9] Gross, A. and Latecki. L.; *"Towards non-parametric digital shape representation and recovery"*; Object Representation in Computer Vision; Proceedings of the International NSF-ARPA Workshop; New-York, USA; Hebert et al.(Eds.); Springer-Verlag; pp.315-325 (1995).

[10] Jakubowski, R.; *"Decomposition of Complex Shapes for their Structural Recognition"*; Information Sciences; **50** (1990).

[11] Janssens, D. and Rozenberg, G.; *"On the structure of Node-Label-Controlled Graph Languages"*; Information Sciences, **20** (1980).

[12] Janssens, D. and Rozenberg, G.; *"Restrictions, extensions and variations of N.L.C Grammars"*; Information Sciences, **20** (1980).

[13] Klauck, C. and Mauss, J.; *"Feature recognition in computer integrated manufacturing"*; Integrated Computer-Aided Engineering; **Vol. 1**, No. 5 (1994).

[14] Montanari, U.G.; *"Seperable Graphs, Planar Graphs and Web Grammars"*; Information and Control; **Vol. 16** (1970).

[15] Pfaltz, J.L. and Rosenfeld, A.; *"Web Grammars"*; Proceedings International Joint Conference on Artificial Intelligence; Washington D.C. (May 1969).

[16] Pratt, T.W.; *"Pair Grammars, Graph Languages and String-to-Graph Translations"*; Journal of Computer and System Sciences; **5** (1971).

[17] Rosenfeld, A.; *"Pyramid Algorithms for Finding Global Structures in Images"*; Information Sciences; **50** (1990).

[18] Van der Walt, A. and Ewert, S.; *"Random Context Picture Grammars"*; Publicationes Mathematicae; **54** (1999).

[19] Wang, P.S.P.; *"Three-Dimensional Object Analysis by Sequential/Parallel Array Grammars"*; Informatique, No 91-7; Centre Scientifique et Polytechnique; Universite Paris Nord; Villetaneuse; France (November 1991).

[20] Wang, P.S.P.; *"A Formal Parallel Model for Three-Dimensional Object Pattern Representation"*; Technical Report NU-CCS-92-11; College of Computer Science; Northeastern University; Boston, MA; U.S.A. (November 1992).