# CDMTCS Research Report Series

# Supplemental Abstracts for DMTCS'01

**C. S. Calude, M. J. Dinneen and S. Sburlan (Editors)**
University of Auckland, New Zealand
"Ovidius" University, Romania

CDMTCS-152
June 2001

Centre for Discrete Mathematics and
Theoretical Computer Science

## Introduction

These are the abstracts of the poster talks to be given at the 3rd International Conference *Discrete Mathematics and Theoretical Computer Science, 2001* to be held at the "Ovidius" University, Constanţa, Romania, on July 2–6, 2001.

# Maximal Complete Bipartite Subgraph Enumeration and Recognition of Association Rules in Databases

Gabriela Alexe, Sorin Alexe, and Peter L. Hammer
RUTCOR, Rutgers University Center for Operations Research, Piscataway, New Jersey, USA

**Abstract.**
Finding association rules in data sources such as on-line databases or data warehouses plays a major role in data mining, e.g. for the identification and classification of business trends. The problem of finding association rules is closely related to that of finding all maximal complete (not necessarily induced) bipartite subgraphs of a graph, a problem which is known to be intractable. We shall present a "consensus" type method, and a lexicographic enumeration method for the determination of all maximal complete (not necessarily induced) bipartite subgraphs of a graph, and compare them with adaptations of known algorithms, using some real-life databases.

# Accelerated Algorithm for Pattern Detection in the Logical Analysis of Data

Sorin Alexe and Peter L. Hammer
RUTCOR, Rutgers University Center for Operations Research, Piscataway, New Jersey, USA

**Abstract.**
A frequently occurring problem in data analysis is that of finding a system of assigning positive or negative "outcomes" to any real $n$-vector, in a way consistent with known assignments to a given subset of such vectors. In the methodology called Logical Analysis of Data (LAD), such a classification method is proposed, using a set of "rules" or "patterns" learned from the assignments of the given data points. The derivation of these patterns was based originally on a "binarization" technique which associated to each real valued variable several binary ones. We propose in this paper a new data structure and an accelerated system of pattern detection. The results of computational experiments indicate major computational advantages of the accelerated pattern detection technique, which will be shown to require approximately the square root of the time required by the standard "naive" algorithm.

# P-Systems and Context Free Languages

Adrian Atanasiu
Faculty of Mathematics, Bucharest University, Bucharest, Romania

Carlos Martin-Vide
Rovira i Virgili University, Tarragona, Spain

**Abstract.**
$P$-systems are computing models, where certain objects can evolve in parallel into a hierarchical membrane structure.
These membranes cannot preserve any order between their own objects. So, if we wish to treat their content as a sequence of characters, then any order will be correct. Thus, languages as $\{\alpha|\alpha \in \{a,b\}^+, \#_a(\alpha) = \#_b(\alpha)\}$ and $\{a^n b^n | n \geq 1\}$ for example, are considered and treated identical.
A construction which preserves the order of the characters in the words for context-free languages is presented in this paper. It is made in $P$-systems with active membranes, where all evolution rules are context-free.
Some preliminary decision problems concerning the words of context-free languages generated with this type of $P$-systems are considered; namely, it can be checked if there are sequences of a given length, or if two sequences are equal or not.

# Symmetry and Antisymmetry in Strings and Sequences

Marian Alexandru Baroni
Liceul CFR Galaţi, Galaţi, Romania

**Abstract.**
For a string of symbols of an alphabet $V$, we examine the symmetry and the antisymmetry of a subset of $V$ with respect to given positions in the string. Languages containing symmetric strings and their corresponding languages related to antisymmetry have the same type in Chomsky hierarchy.
We will also investigate certain languages containing strings which can be extended to bi-infinite symmetric or antisymmetric sequences.

# Locally, Locally Threshold and Piecewise Testable Languages. Some Implemented Algorithms

D. Belostotski, D. Kravtsov, A. Shemshurenko, M. Sobol, A.N. Trahtman and Sh. Yakov
Bar-Ilan University, Dep. of Math. and CS, Ramat Gan, Israel

**Abstract.**
We implement a set of procedures for deciding whether or not a language given by its minimal automaton or by its syntactic semigroup is locally testable, threshold locally testable, strictly locally testable, or piecewise testable. The level of local testability is also found. Some new effective polynomial time algorithms have been implemented as $C^{++}$ package (TESTAS).
Piecewise testable and locally testable languages are the best-known subclasses of star-free languages. These classes with generalizations define two important directions in investigations of these languages.

A locally testable language $L$ is a language with the property that for some nonnegative integer $k$, called the order or the level of local testability, whether or not a word $u$ is in the language $L$ depends on (1) the prefix and suffix of the word $u$ of length $k-1$ and (2) the set of intermediate substrings of length $k$ of the word $u$. Local testablility has a wide spectrum of applications. For instance, regular languages and picture languages can be described with help of a strictly locally testable languages. We implement in our package a polynomial time algorithms of order $O(n^2)$ for local testability problem of automaton and of syntactic semigroup of the language and for finding the order of local testability for syntactic semigroups [Ts].
The locally threshold testable languages generalize the concept of locally testable language. Stochastic locally threshold testable languages, also known as $n - grams$ are used in pattern recognition, particular in speech recognition. We use in our package a polynomial time algorithm for the local threshold testability problem.
A language is piecewise testable iff its syntactic monoid is $J$-trivial (meaning that distinct elements of monoid generate distinct ideals). We implement an algorithm to verify piecewise testability of automaton and of syntactic semigroup of the language (both of of order $O(n^2)$).
The input of the semigroup algorithms is Cayley graph. Maximal size of semigroups we consider was some thousands.

**References:**
[Ts] A.N. Trahtman, A polynomial time algorithm for local testability and its level. *Int. J. of Algebra and Comp.* v. 9, **1**(1998), 31–39.

## Approximate Minima, Epigraphs, and Sections of Functions in Constructive Mathematics

Douglas Bridges and Gabriela Popa
Department of Mathematics & Statistics, University of Canterbury, New Zealand

**Abstract.**
It has been known since the time of Brouwer that although a uniformly continuous function on a compact—that is, complete, totally bounded—metric space has an infimum, it cannot be proved constructively[1] that the infimum is attained (is a minimum). It seems hard to produce good conditions that ensure the attainment of such an infimum. We show that the proof of a theorem of Ekeland ([Aubin], pages 15–16) can be adapted to yield the following result on the existence of strong forms of approximate minima.
**Theorem:** Let $f$ be a uniformly continuous mapping of a compact metric space $X$ into the nonnegative real line $\mathbf{R}^{0+}$, let $x_0 \in X$, and let $\varepsilon, \delta > 0$. Then there exists $\xi \in X$ such that $f(\xi) + \varepsilon\rho(\xi, x_0) \leq f(x_0)$ and $f(\xi) \leq f(x) + \varepsilon\rho(\xi, x) + \delta$.for all $x \in X$.
Let $f$ be a partial mapping of a metric space $X$ into $\mathbf{R}$, and write dom$(f)$ to denote the domain of $f$. For each $\lambda \in \mathbf{F}$ we define the corresponding

- **strict lower section** $S^{\mathrm{sl}}(f, \lambda) = \{x \in \mathrm{dom}(f) : \lambda < f(x)\}$ and
- **upper section** $S^{\mathrm{u}}(f, \lambda) = \{x \in \mathrm{dom}(f) : f(x) \leq \lambda\}$

of $f$. The **strict upper section** $S^{\mathrm{su}}(f, \lambda)$ and the **lower section** $S^{\mathrm{l}}(f, \lambda)$ of $f$ are defined analogously.

[1] By ‘constructively’ we mean ‘with intuitionistic logic’. In other words, we are working with constructive mathematics as developed in [BB].

The constructive proof of Ekeland's theorem depends on a version of an important theorem of Bishop ([BB], page 98, Thm (4.9)).
**Theorem:** Let $X$ be a compact metric space, and $f : X \to \mathbf{F}$ a uniformly continuous function. Then for all but countably many real numbers $\lambda$, the upper and lower sections $S^{\mathrm{u}}(f, \lambda)$ and $S^{\mathrm{l}}(f, \lambda)$ are compact, and the strict upper and strict lower sections $S^{\mathrm{su}}(f, \lambda)$ and $S^{\mathrm{sl}}(f, \lambda)$ are totally bounded.
In the second part of the paper we investigate conditions which are equivalent to the (local) total boundedness of strict lower sections. We define the **subgraph** of $f$ to be

$$\mathrm{L}(f) = \{(x, \lambda) : x \in \mathrm{dom}(f), \lambda \in \mathbf{F}, \lambda < f(x)\}.$$

**Proposition:** The following are equivalent conditions on a partial mapping $f$ of $X$ into $\mathbf{R}$.

*(i)* For all $\lambda$ in a dense subset of $\mathbf{R}$, $S^{\mathrm{sl}}(f, \lambda)$ is either locally totally bounded or empty.

*(ii)* $\mathrm{L}(f)$ is locally totally bounded.

*(iii)* For all but countably many $\lambda \in \mathbf{R}$, $S^{\mathrm{sl}}(f, \lambda)$ is either locally totally bounded or empty.

**References:**
[Aubin] J–P. Aubin, *Optima and Equilibria. An Introduction to Nonlinear Analysis*, Graduate Texts in Mathematics **140**, Springer–Verlag, Heidelberg, 1993.
[BB] Bishop, E. and Bridges, D.S., *Constructive Analysis*, Grundlehren der Math. Wissenschaften **279**, Springer–Verlag, Heidelberg, 1985.

# QED vs QD

Cristian S. Calude
Department of Computer Science, University of Auckland, New Zealand

Elena Calude
Institute of Information Sciences, Massey University at Albany, New Zealand

**Abstract.**
Classically, there are two equivalent ways to look at the mathematical notion of proof: a) as a finite sequence of sentences strictly obeying some axioms and inference rules, b) as a specific type of computation. This gives mathematics an immense advantage over any science: any proof is an explicit sequence of reasoning steps that can be inspected at *leisure*; *in theory*, if followed with care, such a sequence either reveals a gap or mistake, or can convince a skeptic of its conclusion, in which case the theorem it is considered proven.
This equivalence has stimulated the construction of programs which perform like *artificial mathematicians. Artificial mathematicians* are far less ingenious and subtle than human mathematicians, but they surpass their human counterparts by being infinitely more patient and diligent.
If a conventional proof is replaced by a "quantum computational proof", then the conversion from a computation to a sequence of sentences may be impossible. The "quantum mathematician" would say "your conjecture is true", but there will be no way to exhibit all trajectories followed to reach that conclusion.
These facts may not affect the essence of mathematical objects and constructions (which have an autonomous reality quite independent of the physical reality), but they seem to have an

impact of how we learn/understand mathematics (which is thorough the physical world). Indeed, our glimpses of mathematics are revealed only through physical objects, human brains, silicon computers, quantum automata, etc., hence, according to Deutsch (1985), they have to obey not only the axioms and the rules of inference of the theory, but the *laws of physics* as well.

## Bounded Deterministic Go-through Automata

Radu Gramatovici
Faculty of Mathematics, University of Bucharest, Bucharest, Romania

**Abstract.**
We consider the go-through automaton as inspired from the recognition of Marcus contextual languages with shuffle. We compare the descriptional complexity of finite languages implemented by a particular case of deterministic go-through automata, called bounded deterministic go-through automata, with the descriptional complexity of the same class of languages implemented by bounded deterministic push-down automata, deterministic finite automata and deterministic finite cover-automata.

## Relations Between the Low Subrecursion Classes

Cristian Grozea
Faculty of Mathematics, Bucharest University, Bucharest, Romania

**Abstract.**
Much work has been done in order to get a single, natural hierarchy of the primitive recursive classes. The most famous hierarchy, Grzegorczyk's $(\mathcal{E}_r)_{r \in N}$, has problems at lower levels, as it is not useful in characterizing the polynomial time computable functions (probably the most important class of computable functions, the class of feasible functions). Therefore, several other hierarchies have been proposed, culminating with the recent results of S.J. BELLANTONI, K.H. NIGGL, where an interesting ranking of the primitive functions is described, leading to two hierarchies, $(\mathcal{PR}_1^r)_{r \in N}$ (defined by means of primitive recursion) and $(\mathcal{PR}_2^r)_{r \in N}$ (defined by replacing primitive recursion with recursion on notation—binary recursion—in the definition of $\mathcal{PR}_1$). As for $\forall r \geq 2, \mathcal{PR}_1^r = \mathcal{PR}_2^r = \mathcal{E}_{r+1}$, $\mathcal{PR}_1^1 = \mathcal{E}_2$, and $\mathcal{PR}_2^1 = P_f$, these two parallel hierarchies succeed to integrate both FLINSPACE (i.e. $\mathcal{E}_2$) and FPOLTIME (i.e. $P_f$) functions with the Grzegorczyk hierarchy, from the elementary level and above. It was already known that $P_f \not\subset \mathcal{E}_2$, and that $\mathcal{E}_2 \not\subset P_f$, provided $P \neq NP$ (R.V. BOOK)
In this paper we prove that the above results cannot be improved assuming $P \neq NP$, i.e. there is no unique hierarchy to contain both $\mathcal{E}_2$ (or $\mathcal{E}_1$, or $\mathcal{E}_0$) and $P_f$, as $P_f$ is not somewhere in between $\mathcal{E}_0$ and $\mathcal{E}_3$. Although both $\mathcal{E}_0$ and $P_f$ are subsets of $\mathcal{E}_3$, we prove that $\mathcal{E}_0 \not\subset P_f$. In achieving this result we show that $SAT \in \mathcal{E}_0$ which is interesting in itself.

# Phase Transitions in Combinatorial Problems and Their (Possible) Connections with Computational Complexity

Gabriel Istrate
Center for Nonlinear Studies and D-2 Division, Los Alamos National Lab, Los Alamos, NM, USA

**Abstract.**
Starting with the experimental work of Cheeseman, Kanefsky and Taylor, phase transitions have been deemed useful in locating "hard" instances of combinatorial problems. While such a connection seems indeed to exist, the extent to which phase transitions have a bearing on a problem's computational complexity is not clear.
In this talk I will present some recent results on phase transitions in variants of satisfiability, and discuss the precise way these results support the existence of the above-mentioned connection.

# Information: What Does it Mean?

Solomon Marcus
Romanian Academy, Mathematics Department, Romania

**Abstract.**
Etymologically, 'information' comes from the Latin INFORMARE: to put in form, to give a form. This qualitative nature of information was followed consequently by the development of post-Darwinian biology and still today remains very important in the sciences of life and of artistic creativity. On the other hand, the idea of information, as it is considered in mathematics, information sciences, physics, etc. paid great attention to the quantitative aspect of information, a tradition which comes mainly from thermodynamics. The itinerary of these two trends involves practically the quasi-totality of fields of knowledge. To a large extent, we are still not able to articulate in a convenient way the qualitative and the quantitative aspects of information.

# Languages, Infinite Words and their Interaction

Solomon Marcus
Romanian Academy, Mathematics Department, Romania

**Abstract.**
The history of infinite words is older than that of formal languages. Their historical sources are to a large extent different and this is perhaps the reason of the fact that these two lines of

development succeeded to bridge only recently. Our aim is to discuss various ways to transfer ideas and results from formal languages to infinite words and conversely.

# Unbounded Operators on Hilbert Spaces in Constructive Mathematics

Bas Spitters
Department of Mathematics, University of Nijmegen, The Netherlands

**Abstract.**
Unbounded operators are a natural and important extension of bounded operators on Hilbert spaces. They occur in most places where bounded operators are used. As unbounded operators are discontinuous, one might be tempted to think that they can not be handled in constructive mathematics [Hellman1993]. If this were the case, it would be a serious problem for the application of constructive mathematics in quantum mechanics, as the unbounded operators $f(x) \mapsto f'(x)$ and $f(x) \mapsto xf(x)$ on $L^2(\mathbf{R})$ play a fundamental role in quantum mechanics.
Although discontinuous functions are easily handled as partial functions, the challenge to give a satisfying theory of unbounded operators in constructive mathematics still stands. Ye developed a theory for *self-adjoint* unbounded operators, containing a constructive proof of the spectral theorem. He used the definition: $T$ is selfadjoint if $T$ is symmetric$(T \subset T^*)$ and $\mathrm{Ran}(T \pm iI) = H$.
The main results we achieved are that, when $T = T^*$, the hypothesis that $\mathrm{Ran}(T \pm iI) = H$ is equivalent to the locatedness of the graph of $T$, that is the distance

$$\rho((x, y), \{(z, Tz) : z \in H\})$$

can be computed for all $x, y \in H$. Second, a bounded operator $T$ has an adjoint $T^*$ iff its graph is located. Finally, locatedness of the graph is a necessary and sufficient condition for an unbounded normal operator to have a spectral decomposition.
These results suggest that locatedness of the graph is a fundamental property of operators.

# Sequential Systems Approximate States Identification Using Sequential Transition Specifications

Anatol Ursu
Computer Science Department, Technical University of Moldavia, Moldavia

**Abstract.**
A *sequential transition specification* is a propositional temporal logic formula of the form $\Theta \wedge \tau_1 \wedge \tau_2 \wedge \ldots \tau_m \wedge \mathcal{R}$, where $\Theta$ is an initial condition which is a formula of the form $\neg Pr\ \mathcal{U}\ \theta$, $\{\tau_1, \tau_2, \ldots, \tau_m\}$ is a set of transition formulas of the form $\Box(\xi \rightarrow \bigcirc(\neg Pr\ \mathcal{U}\ \eta))$, $\mathcal{R}$ is a restriction, which is an invariant formula of the form $\Box r$, where $r$ is a propositional formula, $\theta, \xi, \eta$ are

disjunctions of history formulas of the form $x_1\mathcal{S}x_2\mathcal{S}...\mathcal{S}x_m$ or $x_1\mathcal{B}x_2\mathcal{B}...\mathcal{B}x_m$, defined on a set of propositional variables $\mathcal{P} = \{x_1, x_2, \ldots, x_n\}$ called input/output vectors, $Pr = \vee_{x_i \in \mathcal{P}} x_i$. This definition referes to temporal operators à la Manna-Pnueli:$\Box$ -*always*, $\mathcal{U}$ – *until*, $\bigcirc$ -*nexttime*, $\mathcal{S}$ -*since*, $\mathcal{B}$ -*back-to*.

To obtain a finite state machine of a sequential system (circuit) using sequential transition specifications, the designer has to: (a) identify the I/O vectors $\{x_1, x_2, x_3, ..., x_n\}$ of the system; (b) specify the behaviors of the system in terms of sequences of I/O vectors. In fact, sequential transition specifications specify temporal equivalence classes, which are sets of sequences of I/O vectors that produce the same sets of next state I/O vectors. An important property of temporal equivalence classes consists in the possibility to approximate the behavior of a sequential system by a set of sequences that is an upper approximation of the concrete set of sequences and to perform successive refinements. Successive approximations of the system behavior can be obtained replacing in the specification the formulas $x_1\mathcal{S}x_2\mathcal{S}\ldots\mathcal{S}x_m$ and $x_1\mathcal{B}x_2\mathcal{B}\ldots\mathcal{B}x_m$ by formulas $x_1\mathcal{S}x_2\mathcal{S}\ldots\mathcal{S}x_p$ and $x_1\mathcal{B}x_2\mathcal{B}\ldots\mathcal{B}x_p$ in the transition program, where $p < m$. Such an approximation is justified by the inclusion $[\![x_1\mathcal{S}x_2\mathcal{S}\ldots\mathcal{S}x_m]\!] \subseteq [\![x_1\mathcal{S}x_2\mathcal{S}\ldots\mathcal{S}x_p]\!]$ and $[\![x_1\mathcal{B}x_2\mathcal{B}\ldots\mathcal{B}x_m]\!] \subseteq [\![x_1\mathcal{B}x_2\mathcal{B}\ldots\mathcal{B}x_p]\!]$.

For example, to check the satisfiability of the transition specification given in the left collumn of the table presented below, we approximate first this transition specification by the specification as presented in the right collumn of the table.

| | |
|---|---|
| $\neg Pr\ \mathcal{U}\ (x_1 \vee x_2 \vee x_3 \vee x_5);$ | $\neg Pr\ \mathcal{U}\ (x_1 \vee x_2 \vee x_3 \vee x_5);$ |
| $\Box((x_3\mathcal{S}x_1 \vee x_4\mathcal{S}x_3 \vee x_5) \to \bigcirc(\neg Pr\ \mathcal{U}\ (x_1 \vee x_3)));$ | $\Box((x_3 \vee x_4 \vee x_5) \to \bigcirc(\neg Pr\ \mathcal{U}\ (x_1 \vee x_3)));$ |
| $\Box((x_2 \vee x_3\mathcal{S}x_2 \vee x_4) \to \bigcirc(\neg Pr\ \mathcal{U}\ (x_1 \vee x_4)));$ | $\Box((x_2 \vee x_3 \vee x_4) \to \bigcirc(\neg Pr\ \mathcal{U}\ (x_1 \vee x_4)));$ |
| $\Box((x_1\mathcal{S}x_5\mathcal{S}x_2 \vee x_2) \to \bigcirc(\neg Pr\ \mathcal{U}\ (x_1 \vee x_3 \vee x_4)));$ | $\Box((x_1 \vee x_2) \to \bigcirc(\neg Pr\ \mathcal{U}\ (x_1 \vee x_3 \vee x_4)));$ |

This transition specification is unsatisfiable. Thus the initial transition specification is also unsatisfiable since we checked the satisfiability of an upper approximation of the initial transition specification. And since this transition specification contains less temporal operators, that is it has a smaller length, the satisfiability analysis is also simpler.

Although the transition specifications are dedicated to approximate rather than to describe precisely the behavior of the sequential systems, they have several important applications in the sequential systems design: (a) since it is relatively simply to write a transition specificatioins, a sequential behavoiur of the system can be easily obtained, (b) a deterministic Büchi automaton can be easily extracted from the transition specifications, (c) transition specifications allow successive approximations of the sequential systems behavior to be obtained, which can be used in stepwise refinement design. For the proposed sequential transition specifications a satisfiability analysis prototype tool **pttf** has been implemented which takes a transition specification and extracts an deterministic approximate Büchi automaton, which accepts all the sequences which satisfy the specification. The proposed sequential transition specifications can be used in approximate sequential system states identification. The approximate finite state machines can be used to fasten the sequential systems verification process.

## Mathematics in the Financial World

Nicholas Dudley Ward
Department of Mathematics, University College, London, UK

**Abstract.**
In this note we give a somewhat critical survey of some mathematical models which are used

in the city (of London) in the banking and insurance markets–and show how one can misuse mathematics in order to make money. We shall argue that most of the models which have brought some people fame and much fortune are on the whole quite vacuous. Continuing in this cynical mode I will trace the roots of my critique back to the production line method of teaching students now prevalent in UK universities.

# A High-Level Programming Language for Image Processing Using Mathematical Morphology

Eugen Zaharescu
Department of Computer Science and Numerical Methods, "Ovidius" University of Constantza, Constantza, Romania

**Abstract.**

***Overview On Grey-Level Image Processing Using Mathematical Morphology***

Mathematical morphology offers a coherent and evolutionary environment for the image analysis and processing. It has an important collection of functional primitives created with some spatial or geometrical methods.
This paper presents first some of the basic functional primitives of the mathematical morphology, used in the analysis of the gray level images. All these functional primitives are created from one basic morphological primitive, using the duality principle and the numerical primitives sup and inf.
Subsequently, the functional primitives ***erosion*** and ***dilation*** are presented. They are given, both, set theory and functional definition and an geometrical interpretation. Also, their topological and algebraically properties are revealed.
This overview continues with the presentations of the derived functional primitives: ***opening*** and ***closing***. The ***morphological erosion*** and ***dilation*** are not each other inverse transforms but dual transforms. Therefore, we can compose them in order to obtain new transforms like the ***morphological opening*** and ***closing***. They are given, subsequently, functional definitions and geometrical interpretations. In the end, several important properties like increasing, anti-extensivity and idempotence are revealed.

***MAX - A High-Level Programming Language for Mathematical Morphology***

The programming language **MAX** (**M**orphology **A**nd e**X**perimentation) has been developed as a very simple language in the style of ***Pascal*** and ***Modula***. Its sole purpose is to evaluate morphological expressions.
***The statements.*** Max has six different types of statement, designed to allow a great deal of flexibility in what kinds of morphological operations can be easily implemented:
***The conditional statement:*** **if** *<expression>* **then** *<statement1>* [ **else** *<statement2>* ];
***The iterative statement:*** **loop** *<statement>[;<statement>...]* **end;**
***The conditional exit statement:*** **exit** $N$ **when** *<expression>*;
***The evaluation statement:*** **do** *<expression>*;
***The output message statement:*** **message** *<expression>*;
***The assignment statement:*** *<variable>* := *<expression>*;

***The variables, the constants and the data types.*** The variables can be declared by stating the data type name followed by a list of variables: *<data_type_name>* *<variable>* [, *<variable>*]**;.** MAX recognize only three data types: **image, pixel, int.**

## Some Legal MAX Binary Operators

**Operator Description**
++ Dilate LEFT by RIGHT
−−Erode LEFT by RIGHT
<< Read an image from a PBM format file named by the string
>> Write an image to a PBM format file named by the string
− > Translate image by the pixel
< − Translate image by the pixel
@ Membership: is the pixel in the image?

```
// EXAMPLE: MAX Program to perform a dilation using operator ++.
image a,b;
begin
do (a<<"$1")++(b<<"$2")>>"$3";
end;
```

**References:**

Serra J.- *Image Analysis and Mathematical Morphology,* Academic Press, London 1982.
Preteux Françoise - *Elements de Morphologie Mathematique: Approche Fonctionelle pour Images Numeriques,* These d'Etat et Sciences, Universite de Paris, Octobre 1987.
Gonzales Rafael, Woods Richard - *Digital Image Processing,* 1992
Jain,A.K.-*Fundamentals of Digital Image Processing*-1989
Parker J.R.-*Algorithms for Image Processing and Computer Vision,* John Wiley & Sons Inc., 1997