Exploring Syntactic Methods for Spatial and Semantic Analysis of Node-Link Diagram Images

Olusola Babalola *

Posted Date: 23 October 2025

doi: 10.20944/preprints202510.1853.v1

Keywords: diagram syntax; constraint multiset grammars; graphics recognition; visual languages

 doi:10.20944/preprints202510.1853.v1

Article

Exploring Syntactic Methods for Spatial and Semantic Analysis of Node-Link Diagram Images

Olusola Babalola

Elizade University, Ilara Mokin, Nigeria; sola@cute.name

Abstract

Diagram image interpretation involves analyzing images of visual representations to extract meaningful information. Since the visual arrangement of diagram elements infers certain semantic concepts specific to a domain, a complete interpretation requires using domain knowledge. Subsequently, a diagram interpretation system must handle the construction of meaning based on this knowledge. Literature from the graphics recognition field suggests grammars from the visual language theory field are useful for encoding domain knowledge and also for the spatial parsing of diagram images. However, previous works failed to suggest - with appraisal - appropriate grammar formalism for a diagram interpretation problem. By examining various characteristics of potential grammar formalisms, we established that Constraint Multiset Grammars (CMG) have sufficient features for handling the diagram analysis task and for constructing meaning from extracted diagram elements according to notational conventions. We also show the requirements of the interpretation task for node-link diagram images; in particular the spatial and semantic analysis stages, and highlight how CMG suffice for handling them.

Keywords: diagram syntax; constraint multiset grammars; graphics recognition; visual languages

1. Introduction

THREE key operations are often carried out if a diagram image is to be automatically interpreted by a machine. The visible graphical objects present in the image must be recognized, the spatial organization of detected graphical objects must be determined, and an interpretation obtained for the various relationships existing between graphical objects; a process dependent on the syntax of the particular diagram notation [1]. The first operation can be referred to as symbol recognition, while the second and third can be referred to as symbol-arrangement analysis. Understanding a diagram requires both the symbol recognition and symbol-arrangement analysis stages [2].

At a symbol recognition stage, the graphical objects in the diagram image are determined using image processing techniques and their locations read from the image. The locations are needed to compute the spatial relationships between graphical objects. Spatial layout analysis is crucial in diagram recognition – especially at the symbol-arrangement analysis stage – and its computation becomes non-trivial as the number of objects in a diagram increases. The spatial analysis challenge is further compounded by diverse spatial relationships that may exist between objects. Therefore an appropriate mechanism for undertaking the spatial layout analysis task is desirable. Valid arrangements of diagram objects vary by diagram type thereby necessitating the use of domain knowledge in carrying out spatial arrangement analysis. One way to involve diagram notation knowledge in diagram interpretation is through the use of grammars and their associated production rules. Grammars are also applied for the spatial parsing process.

Assuming a diagram is correctly segmented and the graphic objects in it correctly recognized the task of interpretation is not yet complete. What is often obtained after recognition is a list of primitives or sometimes higher graphic objects as the case may be, their geometric attributes, and the coordinates of their various boundary boxes. The information obtained from raw diagram pixels

 doi:10.20944/preprints202510.1853.v1

thereafter undergoes processing to yield information at a higher level of recognition. The fact that standards and rules are adhered to in the production of diagrams of a particular notation, enables the use formal grammars to represent the syntax of a diagram notation, and applying the grammar to "parse" the information obtained from the interpretation at the symbol-recognition phase.

One class of knowledge that recognition systems require is knowledge of the diagram notation being examined [3]. For instance, knowledge about the diagram notation can be formalized as grammars making it computable. Knowledge such as the types of primitives (or graphic objects) used in producing a diagram, their visual attributes, and valid spatial relations can be specified and processed once they are expressed as a grammar.

Because grammars can provide a formal definition of the syntax of a diagram notation and form the basis of a parsing algorithm, they are valuable to a diagram recognition and interpretation system. Furthermore, grammars also provide a framework for analyzing semantics of the diagram notation. The semantics of the arrangement of the graphical objects can be determined according to the rules of the particular diagram notation.

Although grammars promise great potentials given the extent of their study in the area of formalizing natural language, in the case of visual languages and naturally occurring visual languages (for example finite state automata diagrams and flowcharts), utilizing grammars to define diagram syntax is challenging. By nature, diagrams are not linear. Diagrams allow several ways to produce, arrange and order elements to communicate meaningfully beyond the adjacency relationship found in most textual languages. This difference reflects in the abstract syntax too, it has been noted that visual representations have directed graphs as their underlying representation rather than trees [2]. Despite this and other challenges that are known to exist with the use of grammars for visual languages, they are widely used to specify visual languages.

To an extent the application of string grammars in the specification of textual languages is standard and well understood. This is unlike the case with diagrammatic languages. A common approach to syntax specification for diagrammatic languages does not exist. Furthermore, no single grammar formalism is guaranteed to satisfy all possible analysis needs of every diagram recognition task. This oftentimes lead to diagram researchers either developing adhoc grammars for their tasks, or finding a formalism that fits the syntactic structures of the diagram being described or that will help them achieve their aims. Examples of researchers that implemented their own grammar formalism include [4,5]. Other works either adapted an existing formalism or simply used it as is.

Our idea is to examine research fields focused on the specification and manipulation of diagrammatic structures to establish grammar options having strong potential as a tool for the syntactic and semantic processing of diagram notations, with reference to the recognition and interpretation of diagram images. Our case study is node-link diagrams, particularly finite state automata diagrams.

Therefore we intend to study from both the graphics recognition and the visual language theory fields, viewing various works from a diagram recognition researcher's perspective. While the objectives of a visual language theory research is likely not the processing of offline diagram images, the plethora of theory, techniques and approaches applied in dealing with the concrete and abstract syntax of visual languages in this field is too promising to be ignored. We must state that we approach the literature as a "user" would seek a tool for their own specific problem, and we therefore make our judgments accordingly.

In the next sections, we examine the problem of spatial and semantic analysis of node-link diagrams and thereafter examine syntactic methods in diagram recognition research and in visual language theory.

2. Defining the Problem of Spatial, Syntactic and Semantic Analysis of Node-Link Diagram Images

A diagram recognition system needs to carry out a significant number of spatial analysis tasks since diagrams use more than just the adjacency ordering common with text. Infact, diagrams are

 doi:10.20944/preprints202510.1853.v1

useful for domains where the information is intrinsically spatial [6]; therefore it can be assumed that diagrams are by nature spatial. Most diagrams are easy to 'read' by humans, making presentation and comprehension of information easier. On the other hand it is currently more challenging for computers to 'read' diagrams compared to reading text.

Diagrams encode meaning visually in symbols and their arrangements. There is a mapping between symbols in a diagram and some particular constructs in the domain of interest. Furthermore, inter-symbol relations, the placement of symbols, and grouping of symbols may also map to constructs. A diagram recognition system must process the arrangement (syntax) and the mappings (semantics).

A. Syntax of Node-Link Diagrams

Graphic and non-graphic elements of a visual representation that must be retrieved from the diagram image for interpretation of a node-link diagram at syntax level, or at a semantic level are listed Table 1 by category. The table covers both individual elements of the diagram, and inter-element arrangements. The meaning of a diagram is obtained by considering all the elements present in a diagram, their respective attributes, and the relationships between symbols (graphical objects). Therefore, starting from a diagram image represented as an array of pixels, we expect to derive or process the various elements in Table 1.

Table 1. Syntactic elements of node-link diagrams.

Category	Elements
A. Symbol - related information (including text labels)	• Symbol type • Symbol attributes • Symbol position • Composite symbols e.g. accepting state, start state • Other visual elements used in the diagram
B. Symbol - relation information	• Acceptable relations • Type of spatial or logical relation existing between symbols • Allowed symbol relations (which type of symbols use which relation)
C. Overall symbol - arrangement system	• Valid symbol positions • Symbol-symbol arrangement pattern • Special symbols and their permitted placement (ordering) e.g. Start symbol in FSA, Start and stop symbol in flowchart • Diagram reading order (if any)

Flowcharts are another type of node-link diagrams. Flowcharts share similar structure with Finite State Automata (FSA) diagrams. Flowchart diagrams are composed of a set of node symbols with a set of connectors connecting a node to another and a set of textual content. The syntactic analysis of flowchart diagrams mostly involves the various elements presented in Table 1.

Ref [7] noted that "a syntactic analysis of flowcharts is a must in order to reach acceptable recognition performances", and suggested that syntactic rules should drive the whole recognition framework.

B. Evaluating Spatial Relations in Node-Link Diagrams

Information such as the bounding box and the centroids of graphic objects realized from an earlier phase of diagram image processing enable computation of various spatial relations that may exist in the diagram. The two properties subsequently form the basis for identifying the spatial relations in a diagram image.

Analyzing the spatial arrangement of a diagram image can be viewed as a search for which graphics objects are arranged in a valid pattern, and which of a set of limited pattern exists between graphical objects whose types and locations are already known. The search must return every correct

 doi:10.20944/preprints202510.1853.v1

spatial relationships exhibited by every graphical object present in the diagram according to the given syntax of the diagram notation.

Spatial analysis is involved at all levels of recognition of syntactic elements of node-link diagrams. This fact can be observed from Table 1. For a start symbol to be identified, spatial relations need to be considered. The same applies to a final state. Obtaining the head or tail region of a directed line (edge) requires some spatial analysis, and to read the complete diagram properly according to the sense of the diagram notation rests primarily on reading valid spatial arrangements.

Since spatial relations are a key aspect of diagrammatic communication, the handling of spatial relations by a grammar formalism is of crucial importance to a diagram recognition researcher; a fluid approach to the treatment of spatial relations is even more important when dealing with offline diagram image recognition.

C. Unique Challenges of the Diagram Recognition Task/Environment

There are challenges encountered during the diagram recognition or interpretation task which may need to be addressed. Noise and ambiguity are common challenges in the graphics recognition problem. Incorrect classification of graphic objects, the outright omission of others as a result of poor digitization or production of the original diagram image are not strange occurrences.

Noise, ambiguity, erroneous segmentation or recognition, all of these problems demands a robust approach if the results of the interpretation process will be of any value.

3. Syntactic Methods in Diagram Recognition and Visual Languages

Over five decades of work in various related and unrelated fields into the form of visual representations resulted in a plurality of grammatical formalisms. While it should be noted that not every of the work was for or about diagrams, the interest ranged from chromosome images to chemical diagrams, technical drawings and even sheet music and dance.

In the attempt to use grammars to specify and describe forms such as the visual which flows in more than one direction, various grammar formalisms and innovative strategies for their use have been discovered. Two trends can be observed in the diverse grammar-based applications: grammars applied to model singular graphical objects; and the modeling multiple graphical objects interacting within a common frame. An example of the latter is the use of a grammar to describe chromosome images, while that of the former is the use of a grammar to describe a technical drawing diagram. It is grammars that can model interplay of graphical objects that are mostly sought in the diagram recognition and interpretation task.

The literature reveals some grammars used for the syntactical analysis of graphical objects and symbols present in images [8–11]. Ref [3] has also provided a brief summary of some applications of syntactic methods in diagram recognition. While specific grammar approaches such as Anderson's grammars, Plex-grammars, stochastic grammars, and shape grammars (among those referenced) are interesting for simple or sometimes complex patterns and shapes, and may be useful for the primitive recognition level of a recognition system, they are not suited for extracting the concrete syntax of complete diagrams or describing diagram notations. However, a number of authors have proposed grammars capable of being used in levels up to the semantic recognition level. We examine those grammar formalisms in the next section.

4. Some Notable Syntactic Approaches in Diagram Interpretation Research

Syntactic methods have been recognized as one of the frameworks for diagram recognition [3]. Other strategies used in diagram recognition system include graph-rewriting, schema-based systems, and the blackboard system. As earlier noted, the visual language theory field does not mainly develop grammars for processing offline or naturally occurring visual languages such as diagrams. This raises the question as to which grammars exist primarily within the graphics recognition and diagram

 doi:10.20944/preprints202510.1853.v1

recognition fields and how grammars have been traditionally used. We mention some selected works in this section.

Formal grammars are variously applied in diagram recognition. They are used in the control of the recognition workflow in the recognition system; in some cases grammars handle the control of the entire recognition system. Grammars are used to store the notation of the diagram convention via rewrite rules, or used in the recognition of specific parts of a drawing for example the dimensions in a technical or engineering drawing [12]. In another approach, different grammars are applied to different tasks or aspects in a diagram recognition system. This can be seen in a recognition system where one grammar formalism is used to represent the visual structure of the diagram and another used for the "logical" representation of the diagram [13]. Ref [12] contains a brief overview of grammars in diagram recognition and of diagram recognition; though the field has advanced, the discussion somewhat remains relevant.

To answer the question of whether there are grammar formalisms that can be said to be "home-grown" in the graphics recognition field, we make mention of two notable diagram recognition systems based mainly on grammars (formalisms developed by the authors themselves) for parsing diagrams of different diagram notations. The Extended Position Formalism (EPF) [4] and Graphics Constraint Grammars (GCG) [5,14,15] were used extensively by their authors in the syntactic interpretation of diagrams from their primitives or their graphical objects.

These two grammar models have been applied to a range of recognition problems in graphics and other visual domains including lawn tennis court recognition, x,y data graphs, finite state automata diagrams, gene diagrams, musical score sheets, tables in old documents. We subsequently explore the two formalisms.

A. Graphics Constraint Grammars (GCG)

Futrelle proposed the GCG and adopted it in solving a variety of recognition problems in various domains. It was used in the diagram understanding system to parse diagrams. In this approach text, polygons and the lines in the diagram are considered the primitives of the grammar system. The grammar for parsing a diagram is made up of rules, with the rules having a set of constraints and a set of productions all of which are specific to a diagram notation. The constraints involve the diagram constituents, the geometric and spatial relations among them, and among set of members, or the overall sets. Constraints can also extend to natural visual concepts such as "all horizontal graphical objects". Propagators express the relationships holding between attributes of a rule object and constituents' attributes.

According to the authors, parsing of GCG is efficient; the grammar makes use of a spatial index which is pre-computed before the actual parsing operation and which is used by the parsing process. Using constraints, grammar rules can be written to effectively cut down on the search space of potential elements. Parsing GCG is more of constraint satisfaction than a classical parsing procedure. The output of the parser is a structure of the diagram (diagram contents) presented in the original diagram image as raw pixels. The content could be represented as a frame-based knowledge representation of the diagram contents. The parsing of line-graphs, finite automata, and some illustrations in biology text have been successfully parsed using the GCG approach.

B. DMOS (Description and Modification of Segmentation)

The creation of DMOS (Description and MOdification of Segmentation) was to enable the analysis of the content of structured documents existing as image pixels. DMOS has been demonstrated for the recognition of tables in documents [16], to recognize musical scores in scanned documents [4], and to tennis courts [17]. The DMOS system is based on the Enhanced Position Formalism (EPF) which was developed for the analysis of images.

EPF is a description language targeted at structured documents enabling a graphical, syntactical, and semantic view or description of a document image. The EPF extends mono-dimensional Definite

 doi:10.20944/preprints202510.1853.v1

Clause Grammar (DCG), adding other operators to the single concatenation operator that is used by the original DCG.

EPF operates on graphic primitives such as line segments and connected components of structures found by lower-level image processing routines in the recognition system. The authors relied on the EPF for their DMOS system which they propose as a generic diagram recognition approach. Once a description of the diagram notation can be provided as an EPF grammar, the grammar is compiled to obtain a new structured document recognition system.

According to DMOS' authors, there was need for a novel grammatical formalism with syntax as simple as possible and with key expressiveness (and context awareness). These qualities were not found in trees and web grammars, plex grammars and graph grammars, the argument of the authors against those being the complex syntax, limited expressiveness, and the need to define production rules as trees or as graphs in certain cases. That was the motivation for the development of DMOS and the EPF formalism.

C. A Look at GCG and EPF According to the Authors and a Review

The following shows part of GCG productions for the Y-axis of a chart.
Y-Axis -> Y-Axis-Line Y-Ticks Y-Labels Y-Text
(:null Y-Ticks Y-Labels Y-Text)
(Y-Axis-Line (Y-Line context))
(Y-Ticks (touch Y-Axis-Line ?)
:constraints (right Y-Ticks Y-Axis-Line))
(Y-Labels (left ? Y-Axis-Line :strip t))
(Y-text (left-nearest ? (or Y-Labels Y-Axis-Line))); [5]

And the following shows a possible EPF grammar production for a rectangle.

rectangle : : =
(segV ---> segLeftSide) &&
AT(touchUp) && segH &&
AT(touchRight) && segV &&
AT(touchDown) && segH &&
AT (touchleft)&4
(segVi--- segleftside) [4].

In Graphics Constraint Grammars, expressiveness is based on constraints. The way GCG use constraints is not strictly or formally defined, therefore it is possible to have relationships and constraints such as *(:constraints (horiz-aligned));* or *(:constraints (vertp Line) (long Line)); and (:constraints (rectanglep Polygon) (small Polygon));*. The scope of looseness is a potential pitfall with this approach. While it seems like some control will be obtained via the spatial index when the parsing process executes, ambiguity is a major issue that we foresee with GCG-based grammars.

In the case of Enhanced Position Formalism grammar, there is a well-defined set of six operators for representing visual constructs. While the authors have shown practical usage of these operators in tackling diagram recognition problems, our opinion is that basing a grammar designed to deal with visual languages on a limited set of operators is a limitation.

The beauty of the two approaches is the fact that they are designed to work from the primitive level and achieve a (semantic) understanding of the diagram within one system or with one grammar. However the challenge of such undertaking is reflected by the need to manage complex issues that will arise in processing diagram semantics from primitive elements. The possibility of grammars getting complicated fast is therefore high.

 doi:10.20944/preprints202510.1853.v1

5. Potential Grammars for the Grammatical Specification of Diagrams with Connections and Geometric Relations

The search for a syntactic approach for a diagram recognition project will lead to research areas like image processing, syntactic pattern recognition, and graphics recognition. These areas are associated with grammar-based applications that will interest a diagram recognition researcher. Although there are grammar-based techniques in the afore-listed areas, many of those techniques may not be suitable for solving the problem of interpreting complete diagrams at a syntactical or semantic level; we earlier noted there are grammars for describing single graphical objects (such as symbols in a drawing) and grammars for an inter-play of graphical objects (such as in the case of a complete drawing involving multiple graphical objects).

For applications targeting the interpretation or analysis of complete diagrams or natural visual languages, we observe that the visual language theory offers more methods and techniques compared to the areas we earlier mentioned. The visual language theory field undertakes a rigorous treatment of schemes for naturally classifying and concisely specifying visual languages [18].

Diagrams can be considered as two-dimensional non-linear representations, and so we attempt to mention some grammar-based works related to them. We understand that it is impossible to make a modest review of what is there in literature on the subject matter in a limited number of paragraphs and therefore refer interested readers to [19,20] for a good coverage of the visual language field. Since syntactic pattern recognition is also concerned with 2D visual structures, we suggest Ref [21] for coverage of syntactic pattern recognition research in general.

One way to narrow down the possible scope when exploring the visual language literature is to examine the aims of an application or work. Visual programming systems, diagrammatic reasoning, and diagrammatic representations are some topics being examined in visual language theory research. But the specification of visual languages seems to be the key area for a diagram recognition researcher to begin their study; [20,22,23] are some literature that treats that aspect of the field.

Refining further what to look at in the wide visual language theory arena may involve considering the problem a diagram recognition researcher needs to solve. The nature of the diagram recognition and interpretation process itself, the sort of input into the system, whether primitives or higher level symbols or geometric shapes, and the characteristics of the diagram notation involved should also determine what works one should include as having potential to be adopted or adapted. Since the primary activity of most visual language investigation is not diagram interpretation, one is also looking for an approach that can fit into the diagram interpretation system.

In our case, the interest is node-link diagrams (using finite-state automata diagrams as a particular instance of that family of visual notation). We therefore limit the discussion to approaches that are generic enough to be used in the treatment of node-link diagrams. By nature, node-link diagrams involve connection and geometric relations between diagram elements. We are therefore interested in Picture Layout Grammars [22], Attributed Multiset Grammars (AMG) [2], Constraint Multiset Grammars (CMG) [24], Graph Grammars, Relation Grammars (RG) [25], Symbol-Relation Grammars (SR grammars) [26], and Extended Positional Grammars [27] since they are able to specify visual languages having properties such as node-link diagram's. Therefore they are potential formalisms for the specification and analysis of node-link diagrams. We exempt grammars which are based on string grammars or their modifications such as Picture Description Language [28] because they are extremely limited by design; using them in the recognition system may not be impossible but the purpose of our investigation is finding the best fit for the task.

Ref [18] is a survey of visual language specification and recognition; different approaches are surveyed and a taxonomy of those approaches is included. A discussion of popular and interesting approaches among those surveyed can also be found in the work. The taxonomy in [18] classifies grammatical approaches to visual language specification into: 1) Generalized string grammars, 2) Graph-grammar based methods, 3) Attributed Multiset Grammars, and 4) Not-in-any-category. The characteristics of some grammars that are members of these classes; especially their key features and limitations will be subsequently examined in this section. Our main goal is not to carry out an

 doi:10.20944/preprints202510.1853.v1

exhaustive analysis but more of appraising approaches for a potential place in the diagram recognition and interpretation system. Essentially, we seek a method for specifying the syntax of a node-link diagram and for extracting meaning from its graphical objects during the diagram interpretation process.

We proceed to examine grammars according to the taxonomy earlier mentioned.

A. Extended Position Grammars (xPGs)

Extended Position Grammars (xPGs) belong to the generalized string languages class, they are an extension to Positional Grammars (PG) which is a context-free grammar formalism. Class diagrams, Petri nets, Statechart diagrams, and Activity diagrams. The productions in xPGs are string-like; during parsing (an LR parsing technique) [29], these productions induce a scanning order on symbols. This method yields an efficient parser for the visual sentences supplied as input.

It is notable that visual constructs specified using xPGs can be parsed efficiently, but the set of visual languages that can be modeled as xPGs is smaller than that of Graph Grammars and Attributed Multiset Grammars families because xPGs are restrictive in the type of visual structures they can successfully describe.

The view of graphical objects in xPGs is as what is termed vsymbols (visual symbols), vsymbols have syntactic attributes and a rule (visual pattern). Our first concern is that the manner xPGs model graphical objects is complicating. A basic example is modeling numbers as a vsymbol having visual pattern matching 0-9. The concept of syntactic attributes in xPGs is also novel in our opinion (considering attaching points for instance). One can say it makes a circle different from a circle depending on whether or not they have the same set of attaching points.

There are unique possibilities that can be achieved with such design in certain levels of diagram recognition. However, xPGs may be challenging for one seeking a grammar to model diagram semantics to handle. The modeling of graphical objects in the diagram to vsymbols is the first hurdle. The style for writing xPGs production rules is the other challenge that needs to be considered. Grammars where the structure of a diagram notation can be easily encoded as production rules are desirable. It is also desirable that a formalism to be used in the analysis of graphical images have features that allow some freedom to introduce considerations as may be required by the grammar writer.

It has been observed that a major weakness of the generalised string grammars family is their limited expressiveness. The authors in [18] stated that the main grammars used for visual language specification are Graph Grammars and Attributed Multiset Grammars. Graph grammars and Attribute Multiset Grammars provide better means of specifying visual languages. We also believe that xPGs is not the best choice for the interpretation and analysis task for most node-link (in particular, FSA) diagram images.

B. Graph Grammars

Comparing graph grammars and Attribute Multiset Grammars in literature shows graph grammars are more popular. This may be due to several reasons; graph grammars are the older of the two, they are based on graph theory which is already well researched with several algorithms and methods and as such is more established, and the range of applications in which graph grammars have been put is impressive, especially in image processing, and graphics recognition, where graph-based applications are quite predominant.

Because of the popularity of graph grammars as a popular tool for syntactic analysis across applications and research disciplines, different graph grammar variants now exist. However, the common consensus is that graph grammars is a powerful formalism for specifying visual languages but has the disadvantage of high cost of processing or parsing them [30]. Ref [31] lists some challenges of parsing graph grammars based formalisms. Since the computational challenge of parsing graph grammars is a well-known problem, various works have made attempt to lower the burden. The Layered Graph Grammar [32] is an example in this regard.

 doi:10.20944/preprints202510.1853.v1

The need for context-sensitivity in graph grammars has also been addressed in certain areas of literature [33]. This is because most graph grammars are context-free yet diagram specification is mainly a context-sensitive enterprise.

However there have been several attempts to work around or limit this disadvantage associated with parsing graph grammars, an example of such is the Layered Graph Grammar [32]. The need for some context-sensitivity while specifying visual languages is also a limitation of graph grammars as most grammars in this family are context-free, this has also been treated in certain areas of literature [33]. Given the extent of work on this topic, one can safely conclude it is possible to have a functional graph-grammar based interpretation system for node-link diagrams.

The usage of graph grammars in the graphics recognition domain is not novel. In diagram recognition, graph grammars have been used in a variety of ways. Ref [34] has observed that majority of applications in mathematics recognition and music notation domains utilize graph transformation.

The authors conjecture in [34] is that graphs are a better choice than multisets for diagram processing, and we find their underlying argument interesting. They considered that explicitly stated spatial relations are better than having to execute complex computation to determine spatial relations. However we differ in our thinking. An explicit stated set of spatial relations will limit the extent of manipulations possible in a diagram recognition system and the process of decoding information locked in pixels, as primitives, symbols, and spatial relations sometimes go beyond the set of explicitly stated spatial relations. In some grammar formalisms, productions can default to external functions to make smarter grammar production rules.

There are varying opinions to the one expressed in [34]. Some authors believe that the initial effort to represent the relationships between terminal symbols that are important and need to be maintained as edges in the initial graph is a disadvantage [18]. Other disadvantages of graph grammars are that large grammars could be intellectually unmanageable [31], noisy and uncertain data are also not handled well [31]. In terms of the style of productions found in graph grammars, the specification of embeddings may be quite complex.

This intense discussion is not only within the research field but also extends to the users requiring a grammar; users are of the opinion that graph grammars are uninviting. Their reasons for this include the steep learning curve and the confusion which may be attendant with the use of graph grammars. Nevertheless graph grammars are key tools used in the graphics/pattern recognition research field and are appealing given what power they possess.

It is clear that the choice of formalism for diagram processing is a contest between graph grammars and Attributed Multiset Grammars (which we discuss in Section C below). The arguments in favor of each of them are important and notable. We proceed to examine Attributed Multiset Grammars.

C. Attributed Multiset Grammars: Picture Layout Grammars, Constraint Multiset Grammars, and Relation Grammars

Attributed Multiset Grammars (AMG) are based on multiset grammars instead of context-free grammars; resulting in productions having an unordered right-hand side. The notion of ordering that exists in string grammars is removed in AMG. Constraints associated with productions specify relationships between graphical objects and also play a role in parsing. In AMG, graphical objects have attributes.

AMG are expressive and as a result of the substantial expressiveness possible with AMG, they cannot be efficiently recognized. Restrictions are therefore introduced to enable and efficient processing of these grammars.

Picture Layout Grammars (PLG) is an attributed multiset grammar which can be efficiently parsed due to restrictions established by the formalism. Efficiency is obtained by limiting the attributes that can be used in production rules. In PLGS, the spatial layout is represented by the attributes.

 doi:10.20944/preprints202510.1853.v1

Constraint Multiset Grammars (CMG) is another member of the AMG class of grammar formalisms. CMG are high-level declarative languages based on multiset rewriting that are capable of being used in the analysis of complex tasks. CMG ability to define complex representations is because CMG rewrite multisets of tokens which may have relationships other than adjacency ordering between them. CMG productions are highly dependent on constraints; constraints are used for specifying the geometric, topological, and spatial relationships present between a collection of objects. Diagram semantics is enforced by a constraint solver. These and a number of other capabilities make diagram analysis easier using CMG compared to most grammar formalisms. Table 2 gives a summary of various CMG features and how they may be used in defining the syntax for a node-link diagram image.

Table 2. Syntactic elements of node-link diagrams and CMG Features.

Diagram Elements	CMG feature that models element
- Symbols or primitives	Diagram primitives and graphical objects (symbols) are easily modeled as terminals (T) and non-terminals (NT) in CMG productions, these have labels and attributes. Diagram elements can also have semantic attributes such as kind (of a particular symbol) or label. A useful feature where there are nodes of different kinds in a diagram.
- Other visual elements used in the diagram - Visual properties of objects	The various geometric properties of a diagram symbol are encoded via geometric and semantic attributes. Attributes of NT (left hand side) are obtained from that of T (right hand side).
- Spatial relations - Composite symbols	- Spatial relations are modeled using geometric attributes of the various graphical objects. - Constraints are computable and define the relationship between a diagram and its components. Negative constraints strengthens interpretation by excluding ambiguity. - External functions can be called to handle any other special needs or logic.
- Diagram syntax	- Symbols have semantic attributes. - Semantic attributes build up the interpretation of diagrams. - Existential quantification to introduce context-sensitivity.

Regarding the efficiency of parsing CMG grammars, it is a known that a practical parser exists, and an algorithm for incremental parsing can be found in [24]. Certain efforts by the grammar writer when composing CMG production rules for a visual language ensures an efficient parsing process. Some of these include ensuring a deterministic grammar, and producing cycle-free productions (avoiding productions that rewrite a non-terminal into another non-terminal). The style of production rules in CMG is fairly straightforward and easy to understand.

By extending PLG, CMG has the ability to parse a larger set of diagrams; generative semantics is also provided by the formalism [24]. A significant number of applications have applied CMG including [36–44]. CMG has been proposed as an analogue of the Chomsky hierarchy for non-linear (visual language) representation.

Relation Grammars are another member of the multiset rewriting class. A visual sentence is viewed as a set of symbol-items and a set of relation-items. Rules are produced for symbols and another set of rules for relations over symbols. Context-free styled rules handle the derivation by rewriting symbol-items and relation-items [26]. A class of SR grammars called the boundary SR grammars has been shown to have an efficient parser as they have lower computational complexity.

6. Choosing a Grammar Formalism for Diagram Interpretation

A primary concern in the choice of grammar for the purpose of analyzing diagram images is the atomic elements (primitives) on which the grammar formalism's production rules are based. This point is closely followed by how spatial relations are realized in the design of the grammar - are there predefined set of spatial operators or by what means are spatial relations achieved? Closely tied to

 doi:10.20944/preprints202510.1853.v1

primitives and spatial relation analysis is how naturally a grammar writer can define the structures and substructures of the diagram notation as production rules. Whatever choice is made, it must be able to sufficiently describe diagram notation and analyze the visual statements during the recognition process. The analysis of the grammar during the recognition process (especially parsing) must also be carried out as an efficient procedure.

The grammar formalisms we have considered so far in this section can handle higher level graphical objects such as geometric symbols, lines, and text as primitives – the graphical objects used in the production of node-link diagrams. The differences between the formalisms discussed in Section V above are the way the primitives and relationships between them are treated, and the restrictions or the freedom applied on primitives and relationships.

Some other weaknesses of these formalisms within a diagram image analysis process may be due to the fact that their essential objective is that they were designed for the processing of formal diagrams produced in controlled environments (for example visual language editors) and not for use in analyzing graphical objects derived from pixels.

We observed that due to the variance between diagram notations, a formalism which may seem weak for a notation may be the perfect option in another case. Despite some inadequacies, the ideas behind these formalisms are valuable when defining diagram notations, specifying syntax or undertaking spatial parsing of diagrams.

Creating an adhoc grammar for analyzing diagram images in recognition systems has a number of downsides. A significant amount of time and effort may be needed to plan and design the grammar; despite the work's main objective not being the creation of a grammar formalism. Furthermore a rigorous treatment or examination of the so-created grammar is unlikely to be as thorough as the various visual language formalisms we have considered – relatively mature and more applied approaches. Finally, it amounts to duplication of efforts in a sense as others have done substantial work in the field that may be adopted and adapted.

EPF and GCG as demonstrated and reported by the respective authors are attractive options for the purpose of analysis of diagrams. The DMOS grammatical method has been used as part of a system to analyze flowchart diagrams in [46]. However like earlier stated, we would prefer a formalism that can handle fairly high-level diagram primitives such as geometric shapes and which provides mechanisms for easily specifying a notation. It should also be possible to seamlessly model a major part of visual languages elements as discussed in [47]. The formalism should also have been rigorously and formally treated. An additional benefit would be if there are sufficient details to conveniently understand and implement them.

Theoretically the specification or description of a visual language can be based on any of the formalisms earlier discussed in Section V; the question is which formalism is the most suitable option for our diagram interpretation task. The motivations for choosing appropriate grammar formalism for modeling a visual notation are influenced by two factors: how easily the grammar formalism describes the language, and the needs of the visual language implementer [48].

One grammar that fits the various requirements is CMG. Aside the common features of terminals and production rules found in grammars, it has additional features; special features such as existential operation, constraints, and error-correction capability. CMG has structures which potentially enable robust handling of diagram recognition tasks and also simplify inclusion of additional undertakings that may be absent in the original design of the grammar formalism.

In summary, CMG is our choice. It was chosen for the analysis of node-link diagram images because it has the core capabilities and supporting features that are needed for the manipulation of the graphical objects and spatial relationships present in node-link diagram images and particular FSA diagram images. CMG has enough expressiveness in its original implementation to describe FSA diagrams without the need for enhancements. Specifying diagrams in CMG is also straightforward, and sufficient information exists in literature to be able to apply CMG in a practical application.

 doi:10.20944/preprints202510.1853.v1

We must point out that adaptations of CMG such as [42], exist, however our analysis, was based on the original features of CMG as presented. We find the original specifications sufficient, working "out of the box" for the analysis of diagrams.

This section is justly concluded by quoting from Wittenburg's paper on visual language parsing [49] concluded *"the field really must agree on standard formalisms (Constraint Multiset Grammars is one candidate) as well as standard representations for input. The continuing proliferation of ill-understood variations on visual language formalisms is not helpful"*.

7. Conclusions

For interpreting the semantics of node-link diagrams according to a specific domain, we came to the conclusion that Constraint Multiset Grammars (CMG) is the best option of a number of alternatives in terms of features for the analysis of node-link diagram images and it can serve as a suitable alternative to developing an ad hoc grammar. CMG do not restrict the scope of allowed spatial relations which can be modeled, and neither is limit placed on the attributes of diagram objects (however they must be computable). This is in sharp contrast to other grammars where possible connection properties are pre-defined. CMG provide a means for describing the syntax of diagram notations. The grammar further allows semantic analysis of the collection of diagram tokens (graphical objects) and their properties as extracted from the diagram image. The problem of adapting string type grammars to analyzing multi-dimensional representations is eliminated by using this formalism.

Features of this grammar formalism are targeted at the needs of those interested in automatic processing of diagrams as pointed out with a case of FSA diagram image interpretation. There is freedom to utilize grammar rules innovatively to meet unique challenges particular to diagram interpretation via those CMG features.

References

1. J. Rekers and A. Schurr, "A graph based framework for the implementation of visual environments," in *Proceedings of the IEEE Symposium on Visual Languages*, 1996, pp. 148-155, 1996.
2. E. J. Golin, "A method for the specification and parsing of visual languages," Tech. Rep. CS-90-19, Department of Computer Science, Brown University, Providence, Rhode Island, United States, 1991.
3. D. Blostein, "General diagram-recognition methodologies," in *Graphics Recognition Methods and Applications* (R. Kasturi and K. Tombre, eds.), vol. 1072 of *Lecture Notes in Computer Science*, pp. 106-122, Springer Berlin, 1996.
4. B. Coüasnon, "DMOS: a generic document recognition method, application to an automatic generator of musical scores, mathematical formulae and table structures recognition systems" in *Proceedings of the Sixth International Conference on Document Analysis and Recognition, 2001.*, pp. 215-220, IEEE, 2001.
5. R. P. Futrelle and N. Nikolakis, "Efficient analysis of complex diagrams using constraint-based parsing," in *Proceedings of the Third International Conference on Document Analysis and Recognition*, 1995, vol. 2, pp. 782-790, IEEE, 1995.
6. B. Chandrasekaran, U. Kurup, B. Banerjee, J. R. Josephson, and R. Winkler, "An architecture for problem solving with diagrams," *Diagrammatic Representation and Inference*, pp. 151-165, 2004.
7. M. Rusiñol and J. Lladós, "Flowchart recognition in patent information retrieval," in *Current Challenges in Patent Information Retrieval*, pp. 351-368, Springer, 2017.
8. K. Fu, "Recent advances in syntactic pattern recognition," in *Pattern Formation by Dynamic Systems and Pattern Recognition* (H. Haken, ed.), vol. 5 of *Springer Series in Synergetics*, pp. 176-185, Springer Berlin, 1979.
9. T. C. Henderson and A. Samal, "Shape grammar compilers," *Pattern Recognition*, vol. 19, no. 4, pp. 279 - 288, 1986.

 doi:10.20944/preprints202510.1853.v1

10. W. F. Miller and A. C. Shaw, "Linguistic methods in picture processing: a survey," *in Proceedings of the December 9-11, 1968, Fall Joint Computer Conference*, Part I, AFIPS '68 (Fall, part I), (New York, NY, USA), pp. 279-290, ACM, 1968.
11. R. Narasimhan, "Syntax-directed interpretation of classes of pictures," *Communications of the ACM*, vol. 9, no. 3, pp. 166-173, 1966.
12. D. Blostein, E. Lank, and R. Zanibbi, "Treatment of diagrams in document image analysis," in *Theory and Application of Diagrams* (M. Anderson, P. Cheng, and V. Haarslev, eds.), vol. 1889 of *Lecture Notes in Computer Science*, pp. 330-344, Springer Berlin, 2000.
13. B. Coüasnon and J. Camillerapp, "Using grammars to segment and recognize music scores," in *Proceedings of DAS'94 IAPR International Workshop on Document Analysis Systems (DAS-94), October 18-20, Kaiserslautern, Germany*, (Kaiserslautern), pp. 15-27, International Association for Pattern Recognition, 1993.
14. R. P. Futrelle, "Strategies for diagram understanding: object/spatial data structures, animate vision, and generalized equivalence," in *Proceedings of the 10th International Conference on Pattern Recognition (ICPR)*, pp. 403-408, IEEE, 1990.
15. R. P. Futrelle, I. A. Kakadiaris, J. Alexander, C. M. Carriero, N. Nikolakis, and J. M. Futrelle, "Understanding diagrams in technical documents," Computer, vol. 25, no. 7, pp. 75-78, 1992.
16. B. Coüasnon, "DMOS, a generic document recognition method: application to table structure analysis in a general and in a specific way," *International Journal of Document Analysis and Recognition (IJDAR)*, vol. 8, no. 2-3, pp. 111-122, 2006.
17. B. Coüasnon, "Dealing with noise in DMOS, a generic method for structured document recognition: an example on a complete grammar," *in Graphics Recognition. Recent Advances and Perspectives* (J. Llados and Y.-B. Kwon, eds.), vol. 3088 of *Lecture Notes in Computer Science*, pp. 38-49, Springer Berlin, 2004.
18. K. Marriott, B. Meyer, and K. Wittenburg, "A survey of visual language specification and recognition," in *Visual Language Theory* (K. Marriott and B. Meyer, eds.), pp. 5-85, Springer New York, 1998.
19. S.-K. Chang, "Visual languages: a tutorial and survey," in Selected *Contributions on Visualization in Programming. 5th Interdisciplinary Workshop in Informatics and Psychology*, (London, UK), pp. 1-23, Springer-Verlag, 1987.
20. K. Marriott and B. Meyer, eds., *Visual Language Theory*. New York, NY, USA: Springer-Verlag New York, Inc., 1998.
21. K. Fu, *Syntactic Pattern Recognition and Applications*. Prentice-Hall Advanced Reference Series: Computer Science, Prentice-Hall, 1982.
22. E. J. Golin and S. P. Reiss, "The specification of visual language syntax," *Journal of Visual Languages & Computing*, vol. 1, no. 2, pp. 141 - 157, 1990.
23. K. Marriott and B. Meyer, "On the classification of visual languages by grammar hierarchies," *Journal of Visual Languages & Computing*, vol. 8, no. 4, pp. 375 - 402, 1997.
24. K. Marriott, "Constraint Multiset Grammars," *in Proceedings of the IEEE Symposium on Visual Languages*, pp. 118-125, IEEE, 1994.
25. C. Crimi, A. Guercio, G. Nota, G. Pacini, G. Tortora, and M. Tucci, "Relation grammars and their application to multi-dimensional languages," *Journal of Visual Languages Computing*, vol. 2, no. 4, pp. 333 - 346, 1991.
26. F. Ferrucci, G. Pacini, G. Satta, M. Sessa, G. Tortora, M. Tucci, and G. Vitiello, "Symbol - relation grammars: a formalism for graphical languages," *Information and Computation*, vol. 131, no. 1, pp. 1 - 46, 1996.
27. G. Costagliola and G. Polese, "Extended positional grammars," in *Proceedings of the IEEE International Symposium on Visual Languages*, 2000., pp. 103-110, 2000.

 doi:10.20944/preprints202510.1853.v1

28. A. C. Shaw, "A formal picture description scheme as a basis for picture processing systems," *Information and Control*, vol. 14, no. 1, pp. 9-52, 1969.
29. G. Costagliola, V. Deufemia, and G. Polese, "Visual language implementation through standard compiler-compiler techniques," *Journal of Visual Languages Computing*, vol. 18, no. 2, pp. 165 - 226, 2007. Selected papers from Visual Languages and Computing 2005 (VLC '05).
30. M. Boshernitsan and M. S. Downes, "Visual programming languages: a survey," Tech. Rep. UCB/CSD-04-1368, Computer Science Division (EECS), University of California, Berkeley, California, 2004.
31. H. Fahmy and D. Blostein, "A survey of graph grammars: theory and applications," in *Proceedings of the 11th IAPR International Conference on Pattern Recognition, 1992. Vol.II. Conference B: Pattern Recognition Methodology and Systems*, pp. 294-298, Aug 1992.
32. J. Rekers and A. Schürr, "Defining and parsing visual languages with layered graph grammars," *Journal of Visual Languages & Computing*, vol. 8, no. 1, pp. 27-55, 1997.
33. D.-Q. Zhang, K. Zhang, and J. Cao, "A context-sensitive graph grammar formalism for the specification of visual languages," *The Computer Journal*, vol. 44, no. 3, pp. 186-200, 2001.
34. D. Blostein, "Dening the syntax and semantics of natural visual languages," in *Applications of Graph Transformations with Industrial Relevance*, vol. 1779 of Lecture Notes in Computer Science, pp. 225-232, Springer Berlin, 2000.
35. D. Blostein, H. Fahmy, and A. Grbavec, "Issues in the practical use of graph rewriting," in *Graph Grammars and Their Application to Computer Science*, pp. 38-55, Springer Berlin, 1996.
36. S. S. Chok and K. Marriott, "Automatic construction of user interfaces from Constraint Multiset Grammars," in *Proceedings of the 11th IEEE International Symposium on Visual Languages*, pp. 242-249, IEEE, 1995.
37. S. S. Chok and K. Marriott, "Automatic construction of intelligent diagram editors," in *Proceedings of the 11th Annual ACM Symposium on User Interface Software and Technology*, pp. 185-194, ACM, 1998.
38. S. S. Chok, K. Marriott, and T. Pato, "Constraint-based diagram beautification," *in Proceedings of the1999 IEEE Symposium on Visual Languages, 1999*, pp. 12-19, 1999.
39. K. Iizuka, J. Tanaka, and B. Shizuki, "Describing a drawing editor by using Constraint Multiset Grammars," in *Proceedings of the Sixth International Symposium on the Future of Software Technology*, 2001.
40. S. Joung and J. Tanaka, "Generating a visual system with soft layout constraints," in *Proceedings of the International Conference on Information (Information'2000)*, pp. 138-145, 2000.
41. S. Macé and E. Anquetil, "Design of a pen-based electric diagram editor based on Context-Driven Constraint Multiset Grammars," in *Human Computer Interaction. Interaction Platforms and Techniques* (J. Jacko, ed.), vol. 4551 of *Lecture Notes in Computer Science, pp. 418-428, Springer Berlin*, 2007.
42. S. Macé and E. Anquetil, "Eager interpretation of on-line hand-drawn structured documents: the DALI methodology," *Pattern Recognition*, vol. 42, no. 12, pp. 3202 - 3214, 2009. New Frontiers in Handwriting Recognition.
43. B. Meyer, "A constraint-based framework for diagrammatic reasoning," *Applied Artificial Intelligence*, vol. 14, no. 4, pp. 327-344, 2000.
44. B. Shizuki, K. Iizuka, and J. Tanaka, "Framework for interpreting handwritten strokes using grammars," in *Computer Human Interaction* (M. Masoodian, S. Jones, and B. Rogers, eds.), vol. 3101 of *Lecture Notes in Computer Science*, pp. 409-419, Springer Berlin, 2004.
45. K. Marriott and B. Meyer, "The CCMG visual language hierarchy," in *Visual Language Theory* (K. Marriott and B. Meyer, eds.), pp. 129-169, Springer New York, 1998.
46. A. Lemaitre, H. Mouchere, J. Camillerapp, and B. Coüasnon, "Interest of syntactic knowledge for on-line flowchart recognition," in *Graphics Recognition. New Trends and Challenges*, pp. 89-98, Springer, 2013.

 doi:10.20944/preprints202510.1853.v1

47. T. Selker and L. Koved, "Elements of visual language," in *IEEE Workshop on Visual Languages*, 1988, pp. 38-44, 1988.
48. G. Costagliola, A. De Lucia, S. Orefice, and G. Tortora, "A framework of syntactic models for the implementation of visual languages," in *Proceedings of the IEEE Symposium on Visual Languages*, 1997., pp. 58-65, IEEE, 1997.
49. K. Wittenburg, "Visual language parsing: if I had a hammer...," in *Multimodal Human-Computer Communication* (H. Bunt, R.-J. Beun, and T. Borghuis, eds.), vol. 1374 of Lecture Notes in Computer Science, pp. 231-249, Springer Berlin, 1998.