مجلة جامعة طرطوس للبحوث والدراسات العلمية_ سلسلة العلوم الهندسية المجلد (3) العدد (٦) 2019
Tartous University Journal for Research and Scientific Studies- Engineering Sciences Series Vol. (3) No. (6) 2019

Predictive adaptive dynamic object's traversals control

Yaroub dayoub*

(Received 18 / 8 / 2019 . Accepted 20 / 10 / 2019)

Abstract

The adaptive dynamic optimal control of an object and selecting next position at multi positions recommended visit depending, so far on the values of it's parameters at last visited position and the passed path and predictively select the next control scenario by using formal grammar.

The registering of the controlled object's parameters and the class belong to the visited position and the passed path it will be required a huge solution table with big searching time, in result using this approach isn't sufficient for solving this problem it's recommended more suitable to use the PREDICATE FORMAL GRAMMAR (PFG)where it's eliminates the randomly selection next position to be visit, which causes to lost extra spend time .

For this reason, it's necessary to predictive select the best next position to be visited.

Using this approach and with a finite set of production rules permits registering object's parameters (consumed time, passed distance, vertex's visit frequency, etc.)at all possible to visit positions, through constructing a set of classes(string series sets), where each class has special set of string series and can generate enough set of powerful string series representing all visited positions by controlled object on the defined class which the constructed string series belongs to).

Keywords: Grammar, production rules, object, class, string, series, path, parameters

*. Assistant, Department of Engineering, Faculty of Technology Engineering of Information and Communication, Tartous University-Syria.

مجلة جامعة طرطوس للبحوث والدراسات العلمية_ سلسلة العلوم الهندسية المجلد (3) العدد (٦) 2019
Tartous University Journal for Research and Scientific Studies- Engineering Sciences Series Vol. (3) No. (6) 2019

التوجيه الديناميكي التنبؤي الملائم لحركة كائن

يعرب ديوب*

(تاريخ الإيداع ١٨ / ٨ / ٢٠١٩ . قُبل للنشر ٢٠ / ١٠ / ٢٠١٩)

الملخص

إن التوجيه الديناميكي الأمثل لحركة كائن ما، واختيار الوضع التالي بين مجموعة نهائية من الأوضاع التي قد يشغلها الكائن الموجه يعتمد الى حد بعيد على آلية بلوغه الوضع الحالي وكيفيته، وماهي قيم بارامتراته في جميع الأوضاع التي شغلها سابقاً والتي يشغلها حالياً، ولتدوين جميع الأوضاع التي مر بها الكائن الموجه والتي تستخدم لاختيار الوضع التالي الواجب شغله، ووفقاً لسيناريو تحكم محدد سابقاً لبلوغ الوضع الهدف، ونظراً لأنه يوجد عدد كبير من الاحتمالات، مما يتطلب معه بناء جداول قد تشغل حجماً ضخماً جداً وذات زمن بحث كبير جدا يصعب معه استخدامها؛ ولذلك نقترح استخدام النحو الشكلي الشرطي PREDICATE FORMAL GRAMMAR لتدوين بارامترات الكائن والتعرف إليه، وبشكل محرفي(سلسلة محارف) .

وبعيداً عن الاختيار العشوائي للموقع التالي الذي يسمح باستبعاد نقل الكائن من وضعية الى أخرى، والذي يؤدي بدوره إلى هدر في زمن التحكم؛ لذا يجب التنبؤ مسبقاً بأفضل المسارات، وعدم اختيار الوضع التالي قبل التأكد من أنه الأفضل. إن استخدام النحو الشكلي يسمح وبمجموعة صغيرة من قواعد الاشتقاق بتدوين الأوضاع التي قد يشغلها الكائن المراد توجيهه، وبشكل سلاسل محرفية تعكس بارامتراته (الزمن ،المسافة المقطوعة او المسار المسلوك...الخ) في الوضع الحالي و جميع الأوضاع التي شغلها سابقاً؛ وذلك بتشكيل أنواع من النحو الشكلي، والتي تولد عدداً كافياً من السلاسل المحرفية لتوصيف الكائن وبدقة متناهية ؛ حيث تخصص كل فئة class بنحو خاص يولّد جميع السلاسل التي قد يشغلها الكائن الموجه، ويتم استخدام قواعد النحو الخاص لتدوين بارامترات الكائن بسلسلة محرفية، وتحديد الفئة التي تنتمي اليها وتوجيهه الى الوضع المتناسب مع سلاسل هذه الفئة.

الكلمات المفتاحية:نحو، قاعدة، كائن، فئة، محرفية، سلاسل، المسار، بارامترات.

* أستاذ مساعد في قسم هندسة تقانة المعلومات – كلية هندسة تكنولوجيا المعلومات والاتصالات –جامعة طرطوس –سورية.

Introduction:

Mainly dynamic optimal object's **traversals** control depending on the visited position and the passed path and then select the next control scenario, using the registered object's attributes as spend time, passed distance, child vertex's frequency visit number, etc.

At recommended graph [1,2] for representation positions which the object can be visit and selecting the next optimal effective control scenario avoiding useless object's traversals, which permits reducing spend time and support dynamic object's scenario control through very complex multistate system, rather than using randomly often useless transferring control between different dialogue class's states, and minimizes dialogue total passed path.

Adaptive object's control efficiency depends so far on supporting adaptive dialogue scenario and solve the main problems as eliminating randomly useless transferring object traversals between different positions depending on the object's parameters at current vertex and the type of passed path which minimizes total spent time, delivery the next Adaptive dynamic dialogue scenario taking in consideration the object's activities history (passed path), smart recording and later recognizing object's activities by using deterministic automats, which defines the next dialogue's scenario class belong to where the controlled object can visit only one vertex (with satisfied predicate) of many recommended to visit (with aren't satisfied predicate), so it will be select next scenario addictively defined.

Object's adaptive control can be get through using linguistic approach using predicate formal grammar "type-1" (non-context-free), it will be used for recording object's parameters and later recognizing object's string series and then selecting the next suitable adaptive control scenario.

Important and aim of the work:

Supporting next adaptive object's optimal control scenario must take in consideration the object's passed path at all visited vertexes (at all different classes) and vertex recently visited by objects. Regarding to the object's parameters (visited child vertex's list, vertex's name and sequence serial number, where each class has special set of child vertexes and spend time, finite string series, etc.), at controlled vertex and providing next control scenario so it can lead the object to the target optimal aimed position by using formal grammar.

Search method and material:

Using this approach at each recommended to visit vertex it's evaluates object's parameters and it will be construct a string series represents the object's parameters at recently visited vertex (denoted by X_{ver}) and history string series (lets denote by x_{hist}) then it will be define the class the controlled objects belong to, and select the next object's control scenario depending on the defined class which permits leading the controlled objects to target final child vertex [3,7,12]. (fig.1), where:

Child vertexes-represent the recommended to visit object's child vertexes, each one has unique name sequence number and class's visiting number.

Arcs- represent possible object's neighbor visiting child vertexes.

Weighted Arcs- represent the distances between neighbor vertexes (or spend time).

At each child vertex evaluated the current object's parameters as character followed by the child vertex's serial number, depending on the current object's state X_{ver} and passed path, it will be constructs string series object' s history series X_{hist} represent the visited child vertexes in different classes and recently visited vertex series X_{ver}, this series will be used for defining the class belong to, so it's possible to visit child vertex belonged to the defined class and eliminates all other unvisited child's vertexes.

For registration set of object's parameters and selecting next suitable control scenario with minimum spent time and passed distance it's useful to use *predicate formal grammar* (pfg,"type-1) [6,8,9,10] defined as follows:

Using this approach permits visiting only one child vertex belonged to an unique class in result no randomly vertex object's visiting of the vertexes.

The predicate formal grammar defines as follows: $\mathbf{G_n=\{V_T,V_N, S, R_{n,m}, P_{n,m}\}}$ where :

$\mathbf{V_t/ V_n}$ -set of terminals/secondary elements respectively.

S-start symbol, $\mathbf{P_{n,m,k}}$ -set of predicates of production rules(n =$\overline{1,c_1}$ - class's sequence serial number(on graph represents row), m =$\overline{1,c_1}$ - production rule's serial number.

k-neighbor serial number (on graph represents column number).

$\mathbf{V_T}$={a,b,c,…,z,0,1,..,9} –set of terminal elements.

$\mathbf{V_N}$={S , $S_{1,1}$,$S_{1,2}$, $S_{1,3}$, $S_{2,1}$, $S_{n-1,m}$ .. S_n }-set of secondary elements.

c_1 , c_2 -unsigned positive integer constants.

$\mathbf{R_{n,m}}$ -set of production rules(n-th -class number, m-serial number) with the following format:

$\mathbf{R_{n,m}}$: $\alpha \xrightarrow{P_{n,m,k}} \beta S_{n,m}$, $P_{n,m}$: $d_{n,m} > c_{n,m}$, $t_{n,m} < c_{n,m}$, $\alpha, S_{n,m} \in V_N$, $\beta \in V_T$ (1)

where:

$d_{n,m}$ - current child vertex's distance at candidate to visit neighbor child vertexes.

$c_{n,m}$ - unsigned integer numeric constants.

$t_{n,m}$ - total consumed time at child vertex $S_{n,m}$.

If and only if the predicate $P_{n,m,k}$ is satisfied then it will be use the related production rule $\mathbf{R_{n,m}}$,so it's possible to registerate the objects parameters at now visited vertex by substitute "α" by "β " then at given child vertex $S_{n,m}$ construct a history string series using the formula x_{hist} +=" β", ($x_{hist}= \epsilon$ *empty string at start position*).

Keeping the left side of production rule without changing and using the related predicates $P_{n,m,k}$, it's possible to construct new production rule with new right site, and in result construct a rich set of production rules, which permits registration object's parameters and control it traversals as follow:

$\mathbf{R_{n,m}}$:$\alpha \xrightarrow{P_{n,m,k}} \mu S_{n,m}$, $P_{n,m,k}$: $d_{n,m} = c_{n,m}$, $t_{n,m} = c_{n,m}$, $\alpha, S_{n,m} \in V_N$, $\mu \in V_T$ (2) .

By comparing relations (1)& (2), using one secondary element (in our condition α)as left site of the production rule it's possible to replace it by N different elements (terminal and nonterminal elements) through using set of different related predicates $P_{i,j}$ consisting in result of N set of string series(language $L(G_i)$,i=$\overline{1,n}$, so each object's visit vertex can belong to a predefined language.

Used one simplified predicate formal grammar for registering and recognizing controlled object permits to construct "n" different formal classes, (each one has it's special set of production rules) which can generate an unique string language $L(\mathbf{G_i})$ Each class has an unique language[9,4,11,12], where :

$$L(G_i) = \{ \{x | x \in V_T^* , x \text{ –string} , S \xrightarrow[G]{*} x\} .$$

Each class has unique language taking in consideration all possible paths and each element reflexes all visited by controlled object states staring at certain start vertex.

The recommended general predicate formal grammar has the following structure (see fig.4

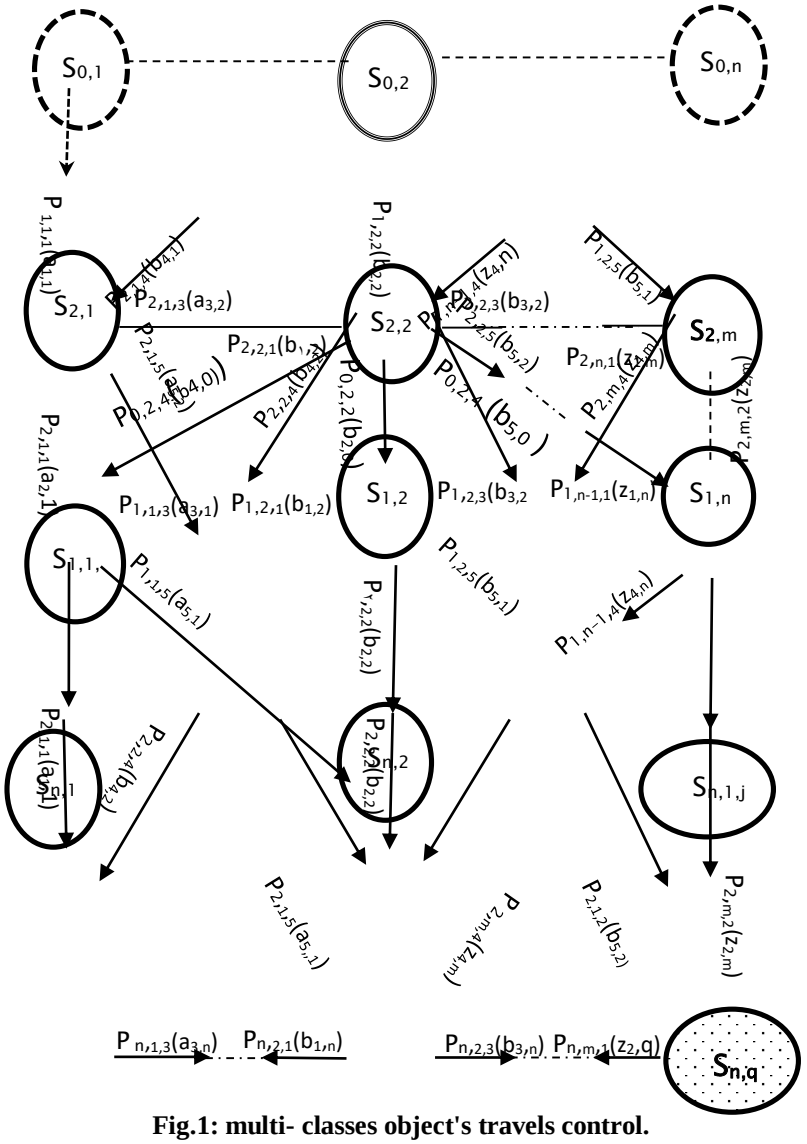


Fig.1: multi- classes object's travels control.
$S_{0,2}$/ $S_{n,q}$- start/end vertexes

Recommended formal grammar defined as $\mathbf{G_n=\{V_T,V_N, S_{0,i+1}, R_{n,m}, P_{n,m,k}\}}$, where: $V_N=\{S_{0,2},S_{1,m+1} ,S_{1,m+2} , S_{1,1+n}, .., S_{2,m+1} , .., S_{2,m+1} \; S_{2,m+c1} , S_{n,m+c2,...} \}$- finite set of

V_T ={a,b,c,z,f,m,n,j,i,0,1,2,..,9} –finite set of terminal elements.

$S_{0,2}$-start element, $R_{n,m}$ -finite set of production rules where n/m-class's /production rules serial number respectively(on the graph represents row /column number, ϵ −empty string) [3,7,9].

$\mathbf{P_{n,m,k}}$ - predicates set of unique production rules, k-neighbor vertexes path serial number. c_i, m, n, k -unsigned positive integer constants.

$R_{n,m}$:	$\mathbf{P_{n,m,k}}$:	
1: $S_{0,m} \xrightarrow{P_{n,m,1}} aNS_{n,m}$	$P_{n,m,1}$: $d_{0,1} >= c_{0,1}$	, $t_{0,1} <= c_{1,2}$
2: $S_{0,m} \xrightarrow{P_{n,m,2}} bNS_{n,m}$	$P_{n,m,2}$: $d_{0,2} > c_{0,3}$	, $t_{0,2} < c_{0,4}$
3: $S_{0,m} \xrightarrow{P_{n,m-1,k}} cNS_{n-1,m}$	$P_{n,m-1,k}$: $d_{n-1,m} = c_{n-1,1}$	, $t_{n-1,m} = c_{n-1,2}$
4: $S_{0,m} \xrightarrow{P_{n,m,k}} zNS_{n-1,m}$	$P_{n,m,k}$: $d_{n,m} = c_{n,2}$	, $t_{n,m} = c_{n,3}$
5: $S_{n,m} \xrightarrow{P_{n,m+1,k}} TNS_{n+1,m}$	$P_{n,m+1,k}$: $d_{n+1,m} > c_{n+1,3}$	, $t_{n+1,m} < c_{n+1,4}$
6: $S_{n,m} \xrightarrow{P_{n,m-1,k+1}} TNS_{n-1,m}$	$P_{n,m-1,k+1}$: $d_{n-1,m} > c_{n-1,5}$	, $t_{n-1,m} < c_{n-1,6}$
7: $S_{n,m} \xrightarrow{P_{n,m,k+2}} TNS_{n,m}$	$P_{n,m,k+2}$: $d_{n,m} > c_{n,7}$	, $t_{n,m} < c_{m,8}$
8: $S_{n,m} \xrightarrow{P_{n,m-1,k+3}} TNS_{n-1,m}$	$P_{n,m-1,k+3}$: $d_{n-1,m} > c_{n-1,10}$	, $t_{n-1,m} < c_{n-1,11}$
9 : $S_{n,m} \xrightarrow{P_{n,m,k}} \epsilon$	$P_{n,m,k}$:finish()=end	
10: $T \xrightarrow{P_{n,10}} a\|b\|..\|z$	$P_{n,10}$:m≠0, n≠0	
11: $n \xrightarrow{P_{n,11}} N\|m$	$P_{n,11}$:m≠N	
12: $N \xrightarrow{P_{n,12}} 0$	$P_{n,12}$: k=0	
13: $N \xrightarrow{P_{n,13}} 1$	$P_{n,13}$: k=1	
……		
24: $N \xrightarrow{P_{n,24}} 9$	$P_{n,24}$: k=9	
25: $N \xrightarrow{P_{n,25}} 0N$	$P_{n,25}$: k>=99	
26: $N \xrightarrow{P_{n,26}} 1N$	$P_{n,26}$: k>=999	
……		
35: $N \xrightarrow{P_{n,35}} 9N$	$P_{n,35}$: k>=m	

Fig.2-structure of Predicate Formal Grammar n^{-th} class

Using production rules $R_{n,m,}$ (Predicate Formal Grammar (PFG) permits generate set of unique powerful string series(languages $L(G_i)$, $i=\overline{1,n}$) enough for registration all passed vertexes at different positions and select next dynamic object's control scenario.

The generated languages $L(G_i)$ will be construct through two steps:

First at current child vertex it will use a certain production rule for registration the child vertex's visiting by character denotes by character "x_{ver}" followed by child vertex's serial number, second construct a history object's string series "x_{hist}" which reflexes all passed child vertexes at different classes.

At each child vertex the constructed string series "x_{hist}" will be used for defining the class belongs to $x_{hist} \in L(G_i)$, $i=\overline{1,n}$, then the controlled object will be support by the control scenario on given child vertex at defined class "i", where all other recommended neighbor

child vertexes will be eliminated, so in result no faulty visiting any neighbor child vertexes and no useless using production rules.

Let's suppose a child vertex "$S_{n,m}$" (n, m- row and column vertex's sequence serial number respectively at graph(see fig.1)has N_i neighbor child vertexes where each one has own character, so for registration object's state at vertexes defined class it's necessary to use Q_i production rules which needs total time $T_i= N_i*t_k*Q_i$, t_k- spend time for visiting one neighbor child vertex, because there aren't useless using of production rules this time will be $T_p= t_k$, as seen $T_i= N_i * Q_i * T_p$, so expression's analysis time becomes very small.

If the related predicate $P_{n,m,k}$ of production rule $R_{n,m}$ is yield, it will be use the corresponding rule $R_{n,m}$ for registration the object's position at recently visited child vertex constructing string $X_{ver,}$ (character followed by a child vertex 's sequence serial number),which will be used for construction x_{hist} taking in consideration all visited before vertexes using concatenation operator "+" , as follows : $x_{hist} += X_{ver}$

Constructed string "x_{hist}" will be used to check if $x_{nist} \in L(G_i)$, $i=\overline{1,n}$, (3,7) if the relation was/ wasn't yield so it will be used to determine and selecting next suitable target object's control scenario.

Each string class must has unique special set of production rules $R_{n,m}$ which can generate unique language(set of string series), $L(G_i)$, $i=\overline{1,n}$, so any string must belong to one language only.

If $x_{hist} \in L(G_i)$, the controlled object must continue at defined class's sequence number "i" starting at next child vertex, where eliminates all other neighbor child vertexes.

In this way, it's possible predictively defining the controlled object's scenario.This permits to accept visiting only one child vertex of candidate vertexes set.

For this reason, it's necessary to define perfectly all classes languages, which will be used for controlling object's traversals through complex positions.

At any child vertex it's possible to select next scenario depending on satisfying the predicate $P_{n,m,k}$ where can be one of the possible following next scenarios as follow:

a) Object at <u>start vertex $S_{0,2}$</u>

For registration the controlled object's state at given vertex "i" it will be use the related production rule $R_{n,m}(S_{0,m} \xrightarrow{P_{n,m,k}} b_{4,0}\ S_{n,m}$, $P_{n,m,k}: d_{n,m} = c_{n,m1}$, $t_{n,m} = c_{n,m2}$, $where\ n = 1, m = 1$) and generates the string "$\mathbf{b_{4,m}}$" (where $\mathbf{b_4}$-path serial number 0-row serial number), t.e momentary string at recently visited child vertex is X_{ver}= "$\mathbf{b_{4,m}}$".

If and only if the conditional expressions $P_{n,m,1}$ was satisfied t.e $d_{0,1} < c_{0,1}$ and $t_{0,1} > c_{0,2}$, so predicate $P_{n,m,1}$ is true and the candidate child vertex to be visit is $S_{1,1}$ on the first class $S_{1,1} \in L(G_1)$.For saving the object's states at all visited child vertexes at different classes "i", it will be use the relation**:**

$X_{hist\ +=} X_{ver} \Longrightarrow X_{hist} =$ ε+"$\mathbf{b_{4,m}}$", (where m=0),because at $S_{n,m}$ history series $X_{hist\ =}\ \varepsilon$ ($\varepsilon-$empty string at start vertex) , $so\ X_{hist}$ ="$\mathbf{b_{4,m}}$" then it will be check if $X_{hist} \in L(G_i)$ $i=\overline{1,n}$, if relation was true the next control scenario will be on the defined "i" class.

Constructed string X_{hist} it will be use to define the class's number belong to by checking the relation $X_{hist} \in L(G_i)$, $i=\overline{1,n}$ if the relation was satisfied then the next control scenario will be continue on the "i" class ,otherwise it will be check if yield this relation $X_{hist} \in L(G_i)$ and next vertexes to visit is $S_{n,m}$(where n=1,m=1).

Depending on the current and history object's state the controlled objet visits child vertex $S_{n,m}$ can get the target position .

1)Visit next child vertex $S_{n,m}$ (n=2,m=1)on the 2-nd class "n=2".

While controlled object visited the new current child vertex $X_{hist} \in L(G_i)$,i=$\overline{1}$, and $X_{hist} = \mathbf{b_{4,m}}$, it will be check the object's parameters $P_{n,m,k}$, t.e $d_{n,m} >= c_{n,1}$, and $t_{n,m} <= c_{n,2}$.

If the predicate$P_{n,m,k}$ is true it will be use the related production rule (see fig.4)

$\mathbf{R_{0,1}}$: $S_{n,m} \xrightarrow{P_{n,m,k}} \alpha\, NS_{n,m}$, **and** $N \xrightarrow{P_{n,k}} 1$, $S_{n,m} \xrightarrow{P_{n,k}} \varepsilon$ ($where\ n = 1,\ m = 2$)

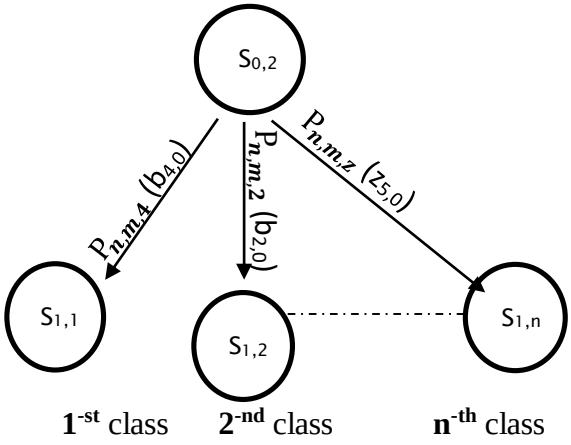


Fig. 3. Object's traversals at start vertex

Generated string series at now visited child vertex is X_{ver}="$a_{3,1}$" $\Longrightarrow$ $X_{hist+=}X_{rver}\Longrightarrow$ X_{hist} ="$\mathbf{b_{4,m}}$"+"$a_{3,m}$" $\Longrightarrow$ X_{hist} ="$b_{4,0}\mathbf{a}_{3,m}$" $\Longrightarrow$if $X_{hist} \in L(G_2) \Longrightarrow$next child vertex to be visit is vertex $S_{n,m}$ (csecond class) by the same way it will be construct any sequence of characters "a".

2) Visit neighbor child vertexes $S_{n,m}$ in the 3-d class.

While controlled object visits the new current child vertex $S_{n,m}$, (where n=1,m=2)and $X_{hist=}$"$b_{4,m}\mathbf{a}_{3,m}$"**, it will be** check the object's parameters $P_{n,m,k}$ are true t.e

$$P_{n,m,k}: d_{n,m} >= c_{n,1}\ , t_{n,m} <= c_{n,2}.$$

If the predicate $P_{n,m,k}$ was true it will be use the related production rule

$\mathbf{R_{n,m}}$: $S_{n,m} \xrightarrow{P_{n,m,k}} \boldsymbol{b}\, NS_{n,m}$, **and** $N \xrightarrow{P_{n,m,k}} 1$, $S_{n,m} \xrightarrow{P_{n,m,k}} \varepsilon$, (where n=1, m=2)

At visited child vertex and by using this production rule, it will be construct the string series, which can be, find by this formula:

X_{ver}="$b_{3,m}$" $\Longrightarrow$ $X_{hist\ +=}X_{ver}\Longrightarrow$ X_{hist} ="$b_{4,0}\mathbf{a}_{3,m}$"+"$b_{3,m}$" $\Longrightarrow$ X_{hist} = "$b_{4,0}\mathbf{a}_{3,m}b_{3,m}$" $\Longrightarrow$if

$X_{hist} \in L(G_n)$, so next child vertex class to be visit is $S_{n,m}$(n=2, m=2).

3) Visit neighbor child vertex $S_{n,m}$ on the n-1 class.

While controlled object visits at the current child vertex $S_{n,m}$ and while the history string series X_{hist} = "$b_{4,0}a_{3,m}b_{3,m}$" **, it will be** check the object's parameters **are satisfied** $P_{n-1,m,k}$: $d_{n-1,m} < c_{n-1,m1}$, $t_{n-1,m} > c_{n-1,m2}$

If the predicate $P_{n,m,k}$ is true it will be use the related production rule:

$\mathbf{R_{1,2}}: S_{n,m} \xrightarrow{P_{n,m,5}} b\ NS_{n,m-1}$, **and** $N \xrightarrow{P_{n,17}} 2$, $S_{n,m-1} \xrightarrow{P_{1,12}} \varepsilon$.

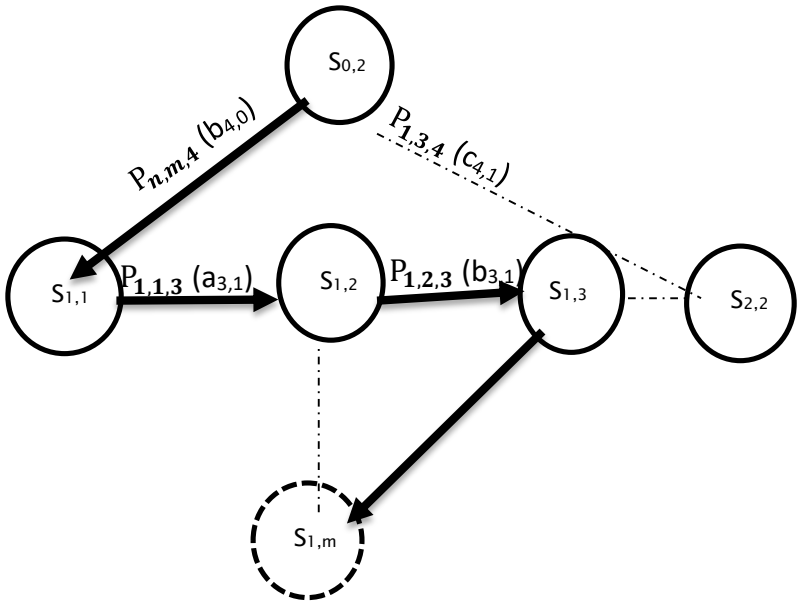


Fig. 4 .passed path $b_{4,0}a_{3,1}b_{3,2}c_{4,1}$

Generated string series at visited child vertex can be X_{ver}= "$b_{4,1}$" $\Longrightarrow$ $X_{hist\ +=}$Xver $\Longrightarrow$ X_{hist} = "$b_{4,0}a_{3,m}b_{3,m}$"+"$b_{4,1}$" $\Longrightarrow$ X_{hist} ="$b_{4,0}a_{3,m}b_{3,m+1}b_{4,m+2}$" $\Longrightarrow$if $X_{hist} \in L(G_2)$ $\Longrightarrow$ next child vertex is on neighbor class to be visit is $S_{n,m}$ **(**where **n=m=2)**.

4)Visit neighbor child vertex $S_{n,m+1}$ in the n+1 class(suppose n=3).

While controlled object visits child vertex $S_{2,1}$ and X_{hist} = "$b_{4,0}a_{3,m}b_{3,m+1}b_{4,m+2}$"**, it will** check the object's parameters **are satisfied** :

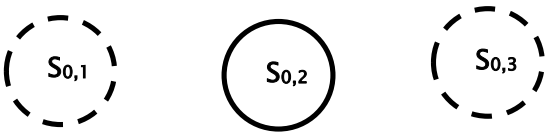
$P_{n,m,k}$: $d_{n,m} < c_{n,m}$, $t_{n,m} > c_{n,m}$), if the predicate $P_{n,m,k}$ is true it will be use the related production rule

($\mathbf{R_{n,m}}$: $S_{n,m} \xrightarrow{P_{n,m,3}} b\ NS_{n,m}$, **and** $N \xrightarrow{P_{n,18}} 3$,, $S_{n,m} \xrightarrow{P_{1,32}} \varepsilon$, (where n=3,m=2)

Generated string series at now visited child vertex can be X_{rver}= "c_3" $\Longrightarrow$ $X_{hist+=}$$X_{ver}$ $\Longrightarrow$ X_{hist} ="$b_{4,0}a_{3,m}b_{3,m+1}b_{4,m+2}$"+"$c_{3,m}$ $\Longrightarrow$ X_{hist} = "$b_{4,0}a_{3,m}b_{3,m+1}b_{4,m+2}c_{4,m}$", then check if string series $X_{hist} \in L(G_4)$ $\Longrightarrow$next child vertex class "n=4" to be visit (suppose the number of classes is 4), by the same way it will be continue any sequence string series.

In the same way and at each vertex it will be defines the next scenario depending on the object's passed path and the object's parameters at visited child vertex consisting in result the string followed with vertex serial sequence number as:

$b_{2,0}$, $b_{2,1}b_{2,2}$, $b_{2,0}b_{2,1}b_{2,2}b_{2,3}b_{2,4}$, .. etc. at given class (series second class) or series at



different classes($a_{1,0}$,$a_{1,1}b_{3,2}c_{3,3}d_{3,3}$, $c_{1,0}$,$c_{2,1}b_{4,2}a_{3,2}b_{5,2}$,..).

If the predicate $P_{n,m,k}$ wasn't satisfied for any class's languages $L(G_n)$,it will be a fatal error and(it's necessary to eliminate this condition because it's leading to stop working) so it means no spend time for analysis an useless faulty visiting any child vertex.

discussion:

Registration object's parameters and predictively selecting next control scenario.

Suppose the number of passed vertexes can classified in 4 categories(classes) and passed path p="$b_{4,0}b_{3,1}c_{4,1}b_{1,2}a_{5,2}b_{3,3}$ $c_{3,3}$" ,where $p \in L(G_i)$, i=2 and the object must to get the vertex $S_{3,4}$ in the given order consists of the following vertices as follows:

V_2={$S_{0,2}$, $S_{1,2}$,$S_{1,3}$,$S_{2,2}$, $S_{2,1}$, $S_{3,2}$,$S_{3,3}$,$S_{3,4}$}- finite list recommended to pass vertexes names.

a/b/c/d- Used characters for object's state registration at 1^{-st},2^{-nd},3^{-d},4^{-th} classes respectively each one tailed with path's number to neighbor vertex (vertex's serial number example: $b_{3,1}$,where 3/1- path's vertex's serial number respectively).

Recommended formal grammar G_2 defined by using global grammar:

G_n ={V_T,V_N, S, $R_{2,m}$, $P_{2,m}$ } [3,7,9]where:

V_N={$S_{0,2}$,$S_{1,1}$,$S_{1,2}$,$S_{1,3}$, $S_{2,1}$, .. ,$S_{2,3}$,$S_{3,1}$,.. ,$S_{3,4}$}- finite set of non-terminal elements.

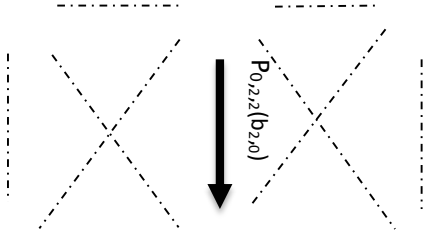
V_T ={ a, b, c, d, 0,1,2,..,9} –finite set of terminal elements.

$S_{0,2}$ -start element, $R_{3,4}$ - finite set of production rules, ϵ −empty string.

$P_{2,m}$- set of production rule's predicates on class "m=2".

n=$\overline{1,n1}$ – class's sequence serial number, m=$\overline{1,m1}$ - production rule's serial number.

m1,n1 –row/column serial number, k-path serial number.



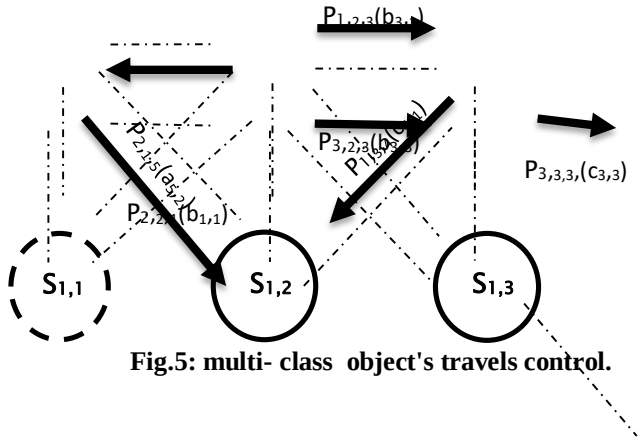


Fig.5: multi- class object's travels control.

$R_{2,m}$: $P_{n,m,k}$:

1: $S_{0,2} \xrightarrow{P_{2,0,2}} bNS_{1,2}$ $P_{2,0,2}$: $d_{0,2} > c_{0,2}$, $t_{0,2} < c_{0,3}$

2: $S_{1,1} \xrightarrow{P_{1,1,3}} TNS_{2,1}$ $P_{1,1,3}$: $d_{1,1} >= c_{1,1}$, $t_{1,1} > c_{1,2}$

3: $S_{1,2} \xrightarrow{P_{1,2,3}} TNS_{1,3}$ $P_{1,2,3}$: $d_{2,2} > c_{2,2}$, $t_{2,2} <= c_{2,3}$

4: $S_{1,3} \xrightarrow{P_{1,3,4}} dNS_{2,2}$ $P_{1,3,4}$: $d_{3,2} = c_{3,2}$, $t_{3,2} > c_{3,3}$.

5: $S_{2,1} \xrightarrow{P_{2,1,2}} TNS_{3,1} \mid \epsilon$ $P_{2,1,2}$: $d_{2,1} > c_{2,4}$, $t_{2,1} < c_{2,5}$

6: $S_{2,2} \xrightarrow{P_{2,2,5}} TNS_{3,3} \mid \epsilon$ $P_{2,2,5}$: $d_{2,6} = c_{2,6}$, $t_{2,6} = c_{2,7}$

7: $S_{2,1} \xrightarrow{P_{2,1,3}} TNS_{2,2} \mid \epsilon$ $P_{2,1,3}$: $d_{1,2} >= c_{1,2}$, $t_{1,2} <= c_{1,3}$

8: $S_{3,2} \xrightarrow{P_{2,2,5}} TNS_{3,3} \mid \epsilon$ $P_{2,2,5}$: $d_{2,3} > c_{2,7}$, $t_{2,3} <= c_{2,8}$

9: $S_{3,3} \xrightarrow{P_{2,2,3}} TNS_{3,4} \mid \epsilon$ $P_{2,2,3}$: $d_{2,1} > c_{2,1}$, $t_{2,1} < c_{2,2}$

10: $S_{3,1} \xrightarrow{P_{2,3,1}} TNS_{2,2} \mid \epsilon$ $P_{2,3,1}$: $d_{3,1} > c_{3,1}$, $t_{3,1} > c_{3,2}$

11: $S_{3,3} \xrightarrow{P_{2,3,3}} TNS_{4,2}$ $P_{2,3,3}$: $d_{3,3} > c_{3,3}$, $t_{3,3} < c_{3,4}$,

12: $S_{3,4} \xrightarrow{P_{2,m}} \epsilon$ $P_{2,3,4}$: $f_n = c_1$ or finish()=end, (ϵ −empty string).

13: $T \xrightarrow{P_{2,1,13}} a|b|..|z$ $P_{2,1,13}$: m≠0, n≠0.

14: $n \xrightarrow{P_{2,1,14}} N \mid m$ $P_{2,1,14}$: m≠n.

15: $N \xrightarrow{P_{2,15}} 0$ $P_{2,15}$: k=0.

16: $N \xrightarrow{P_{2,16}} 1$ $P_{2,16}$: k=1

……

23: $N \xrightarrow{P_{2,1,24}} 9$ $P_{2,24}$: k=9

24: $N \xrightarrow{P_{2,25}} 1N$ $P_{2,25}$: k>=99

25: $N \xrightarrow{P_{2,26}} 2N$ $P_{2,26}$: k>=999

.....

34: $N \xrightarrow{P_{2,34}} 9N$ $P_{2,34}$: k>=m

Fig.6-structure of Predicate Formal Grammar 2-nd class's $L(G_2)$

Suppose the optimal path necessary to pass by the object is started at vertex $S_{0,2}$ and stopped at vertex $S_{3,4}$, through the following vertexes set in the following sequence:

$\{ S_{0,2}, S_{1,2}, S_{1,3}, S_{2,1}, S_{3,2}, S_{3,3}, .., S_{3,4,}\} \in S_{n,m}$, ($n=\overline{1,3}$, $m=\overline{1,4}$).

Object allocates at vertex $S_{0,2}$,it will check if predicate $P_{2,0,4}$ (represent object's parameters and for demonstration purpose we will suppose specified predicates are true) is true:

$d_{0,2} < c_{0,2}$ and $t_{0,2}>c_{0,3}$,because $p_{0,2,4}$ =true it will use the corresponding production rules as follows(fig.6):

$$R_{2,1}: S_{0,2} \xrightarrow{P_{2,0,4}} bNS_{1,2} \quad \text{and} \quad R_{2,15}: N \xrightarrow{P_{2,1,15}} 4$$

Using production rules $R_{2,1}$&$R_{2,15}$ at last visited vertex will generate string series X_{ver} = "$b_{4,0}$" , which specifies the controlled object at vertex $S_{0,2}$ and object's history string series X_{hist}reflects passed path which can be find as follow:

$X_{hist} += x_{ver,}$ $where$ X_{ver} ="$b_{4,o}$" $\Longrightarrow X_{hist}$ ="ϵ" & $X_{ver} \Longrightarrow X_{hist}$ ="ϵ"& "$b_{4,0}$"="$b_{4,0}$"and because $X_{hist} \in L(G_2)$,so the next vertex recommended to visit is $S_{1,2}$, while eliminates visiting all other neighbor vertexes(all predicates in false conditions).

At visited vertex $S_{1,2}$,it will check if $P_{1,2,3}$=true, while this predicate is true t.e $d_{1,3}> c_{1,3}$, $t_{1,3}<c_{2,3}$, so it can use the grammar's production rule $R_{2,3}$ (fig.6):

$$R_{2,3}: S_{1,2} \xrightarrow{P_{2,2,3}} bNS_{1,3} \quad \text{and} \quad R_{2,18}: N \xrightarrow{P_{2,2,18}} 3$$

Using production rules $R_{2,3}$&$R_{2,8}$ will generate string series X_{ver} = "$b_{3,1}$" which specifies the controlled object at vertex $S_{1,2}$ as follow:

while $x_{ver} = "b_{3,1}" \Longrightarrow X_{hist}$ = "$b_{4,0}$" + "$b_{3,1}$"="$b_{4,0}b_{3,1}$", because $X_{hist} \in L(G_3)$, so the next vertex recommended to visit is $S_{1,3}$.

At vertex $S_{1,3}$, object will check if $P_{1,3,4}$($d_{1,3}> c_{1,3}$, $t_{1,3}<c_{2,3}$), while this predicate $P_{1,3,4}$=true so it can use the grammar's production rule $R_{3,4}$.

$$R_{3,4}: S_{1,3} \xrightarrow{P_{1,3,4}} cNS_{2,2} \quad \text{and} \quad R_{2,19}: N \xrightarrow{P_{1,3,19}} 4$$

,so x_{ver} = "$c_{1,4}$", then constructed series X_{hist} = " $b_{4,0}b_{3,1}$"+"$c_{1,4}$" = " $b_{4,0}b_{3,1}dc_{1,4}$", and because $X_{hist} \in L(G_2)$,so the next vertex recommended to visit is $S_{2,2}$.

* At $S_{2,2}$ vertex will check if $P_{2,2,2}$=true($d_{2,2} > c_{2,2}$, $t_{2,2} <c_{2,3}$), while this predicate is true the construction grammar's production rules permits using only $R_{2,5}$.

$$R_{2,5}: S_{2,2} \xrightarrow{P_{2,2,2}} bNS_{2,1} \quad \text{and} \quad R_{2,16}: N \xrightarrow{P_{2,2,16}} 1$$

Using $R_{2,5}$ production rule, so $x_{ver} = "b_{1,2}"$, while X_{hist} = " $b_0b_{3,1}c_{1,4}$", so new value X_{hist} = " $b_0b_{3,1}c_{1,4}$" + = "$b_{1,2}$" $\Longrightarrow$ X_{hist} = "$b_0b_{3,1}c_{1,4}b_{1,2}$" ,because

$X_{hist} \in L(G_1)$, so the next vertex recommended to visit is $S_{2,1}$.

* object allocates at vertex $S_{2,1}$, it will check if $P_{2,1,5}$($d_{2,5} > c_{2,5}$, $t_{2,5} < c_{2,6}$) is true, while this predicate is true the related grammar's production rule $R_{2,11}$ will use.

$$R_{2,11} : S_{2,1} \xrightarrow{P_{2,2,2}} aNS_{3,2} \quad \text{and} \quad R_{2,17}: N \xrightarrow{P_{2,2,20}} 5$$

Using this production rule $R_{2,11}$, so string series at visited vertex is $x_{ver} = "a_{5,2}"$, while

$X_{hist} = " b_{0,4}b_{3,1}c_{1,4}b_{2,2}"$, so new value $X_{hist} = "b_{4,0}b_{3,1}c_{1,4}b_{2,2}" += ""a_{5,2}" \Longrightarrow$ $X_{hist} = " b_{4,0}b_{3,1}c_{1,4}b_{2,2}a_{5,2}"$ because $X_{hist} \in L(G_2)$,so the next vertex recommended to visit is $S_{3,2}$.

* At $S_{3,2}$ vertex will check if $P_{3,2,3}$($d_{3,2} > c_{3,2}$, $t_{3,2} < c_{3,3}$) is true, while it's true the corresponding grammar's production rule ($R_{2,7}$)will be used as follow:

$$R_{2,1} : S_{3,2} \xrightarrow{P_{3,1,3}} bNS_{3,4} \quad \text{and} \quad R_{3,5}: S_{3,2} \xrightarrow{P_{3,1,6}} TNS_{1,2}$$

Using this production, so $x_{ver} = b_{3,3}$, $where$ $X_{hist} = "b_{4,0}b_{3,1}c_{4,1}b_{1,2}a_{5,2}"$, so new value $X_{hist} == "b_{4,0}b_{3,1}c_{4,1}b_{1,2}a_{5,2}" += "b_{3,3}" \Longrightarrow X_{hist} = " b_{4,0}b_{3,1}c_{4,1}b_{1,2}a_{5,2}b_{3,3}"$, and

Because $X_{hist} \in L(G_2)$ an next vertex to visit is $S_{3,3}$.

* At $S_{3,3}$ vertex it will be check if $P_{3,2,3}$($d_{3,2} > c_{3,2}$, $t_{3,2} < c_{3,}$) is true, while it's true the corresponding grammar's production rule ($R_{2,7}$)will be used as follow:

$$R_{2,7} : S_{3,2} \xrightarrow{P_{3,2,3}} cNS_{3,3} \quad \text{and} \quad R_{3,5}: S_{3,2} \xrightarrow{P_{2,3,18}} TNS_{3,3}$$

Using this production rule , so $x_{ver} = "c_{3,3}"$, $where$ $X_{hist} = " b_{4,0}b_{3,1}c_{4,1}b_{1,2}a_{5,2}b_{3,3}"$ and new value $X_{hist} = "b_{4,0}b_{3,1}c_{4,1}b_{1,2}a_{5,2}b_{3,3}" + = "c_{3,3}"$,where object's history string series became $X_{hist} = "b_{4,0}b_{3,1}c_{4,1}b_{1,2}a_{5,2}b_{3,3}c_{3,3}"$,and next vertex to visit is

$S_{3,4} \in L(G_4)$ and because it's the final position ,so the execution will be finished and while $S_{3,4}$ is the last vertex to visit so it will use the production

$$R_{3,12}: S_{3,4} \xrightarrow{P_{3,4,m}} \epsilon \qquad P_{3,4,m}: f_n = c_1 \text{ , finish()=end}$$

As seen the constructed string series specifies the controlled object at given vertex $S_{n,m}$ (taking in consideration all passed vertexes) as string series ,if the predicate $P_{i,j,k}$ was false it will check the next predicate if it's $P_{n+1,m+1,k}$ =true ? , so next vertex recommended to visit is $S_{n+1,m+1}$.

By the same way it will check the other predicates .

It's necessary to mention that the author can consist a set of production rules as he wants and leads the controlled object from one vertex to the target as required.

5.*Effect of using predictively control scenario to the controlled object :

Suppose the number of classes is 4 and the number of vertexes which can visited by controlled object's arranged in 3 rows and 4 columns, where the object can visit each vertex if a certain conditional expression is true (eliminates all other vertexes), while the possibility of vertex's visiting number with/without using predicates of the production rules of the formal grammar reflected here (at tab.1).

As seen from the tab.1 and the fig.7 using predicate formal grammar and predictively diagnose the next controlled object's scenario permits get big gain and save time.

Vertex's Visit serial number	$S_{1,1}$	$S_{1,2}$	$S_{1,3}$	$S_{1,4}$	**$S_{2,1}$**	$S_{2,2}$	$S_{2,3}$	$S_{2,4}$	**$S_{3,1}$**	$S_{3,2}$	$S_{3,3}$	$S_{3,4}$
Normal Visiting number of Vertex(N_1)	**14**	13	13	14	**42**	35	35	42	**42**	35	35	42
Conditional Visiting number of Vertex(N_2)	**4**	2	2	4	**15**	13	13	15	**15**	13	13	15
$\Delta=N_1-N_2$	**10**	11	11	10	**27**	22	22	27	**27**	22	22	27

Tab.1

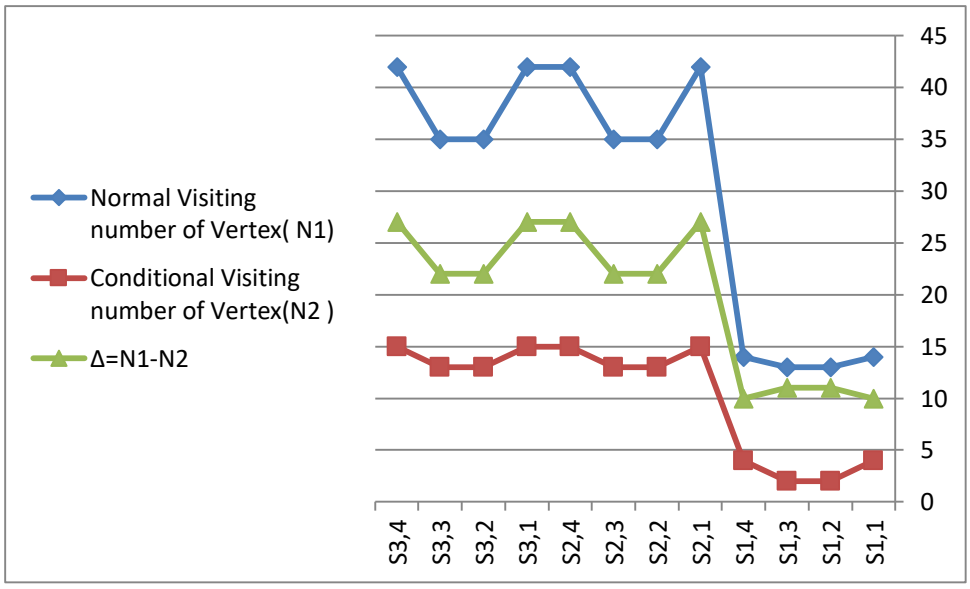


Fig7.Effect of using formal grammar for object's traversals control

Results and conclusion:

Using predictively object's control scenario using formal grammar permits the following:

1-string registration of the controlled object visited different states and specify each one with it's special character.

2-suport wide range for describing the controlled object at different classes by different way.

3-it's easy to define the object's coordination and it's state.

4-predictevely selecting the optimal path regarding to predefined object's parameters so it's possible execute a certain path by different way.

5. Using a few production rules it's possible to describe a very complex object's movement.

6-proposed predicates permits using many production rules in the same context so generate more powerful and rich language enough for registration object's complex traversals and recognizing it's class.

References:

1. Graph Transformation: 12th International Conference, ICGT 2019, Held as Part of STAF 2019, Eindhoven, the Netherlands, July 15–16, 2019, Proceedings (Lecture Notes in Computer Science) Paperback – Import, 25 Jun 2019.

2. An-Bo Li, Ying Chen, Guo-Nian Lü, A-Xing Zhu, Automatic Detection of Geological Folds Using Attributed Relational Graphs and Formal Grammar, Computers and Geosciences (2019).

3. Sandro Schönborn, Bayesian Linear Regression, Pattern Recognition ،university of Basel 2016.

4. Knuth, D. E. "On the translation of languages from left to right" (PDF). Information Information and Control. 8 (6): 607–639. 2011.

5. Dr. G. Englebienne, Machine Learning Pattern Recognition 2018.

6. Alfred V. Aho, Monica S.Lam, Jeffrey. D. Ullman. Compilers principles, techniques&tools, Pearson-Addison Wesley,2007.

7. A. Jain, R. Duin and J. Mao (2000), Statistical Pattern Recognition. A Review IEEETPAMI , Vol. 22, No. 1 (Jan 200)4-37.

8. Hopcroft, J.E., R.Motwani,and J.D. Ullman. Introduction to Automata Theory, Languages, and computation, Addtion-Wesly, Boston MA,2006.

9- Chomsky, N., "Three models for the description of language," IRE Trans. on Information Theory IT-2:3(1956), pp.113-124.

10.guttorp, Peter; MININ, Vladimir N. Stochastic modeling of scientific data. Chapman and Hall/CRC, 2018.

11. Saavedra, Victor, et al. Pattern Recognition Receptors in Auto inflammation. In: Textbook of Auto inflammation. Springer, Cham, 2019. p. 61-87.

12. د.يعرب ديوب." نمذجة المحادثة التعليمية المؤتمتة مع الكائن" مجلة جامعة البعث،2013 المجلد35

13. Stochastic logical linguistic approach Multi-level automated object's dialogue control (MADC), Tishreen institute,2013.