Global Journal of Pure and Applied Mathematics.
ISSN 0973-1768 Volume 11, Number 5 (2015), pp. 3939-3952

http://www.ripublication.com

VIRT Charts To Multiport Component Syntactic Charts Transformation

Vladimir M. Polyakov and Yury D. Ryazanov

The Belgorod State Technological University of V. G. Shukhov,
Russia, 308012, Belgorod, Kostyukov St., 46
The Belgorod State Technological University of V. G. Shukhov,
Russia, 308012, Belgorod, Kostyukov St., 46

Abstract

In article questions of compact representation of the formal languages in the form of syntactic charts are considered. Such representation allows to build translators, efficient on memory. The modified syntactic charts (MSC), and also a way of transformation of Virt charts to MSC are offered. The relation of strict equivalence on a set of the MSC hubs and algorithm of check of accessory of cluster pairs to the relation of strict equivalence is defined. The system of transformations keeping strict equivalence, and a way of obtaining the strict equivalence of the syntactic chart given on the relation is offered. The quantity of components in MSC given on the relation of strict equivalence can be less, than in the equivalent it to Virt's chart, and components, unlike components of Virt charts, can have more than one entrance. The extent of the strict equivalence of MSC given on the relation in general, as a rule, is less than a size of equivalent to it Virt's chart. The example of the chart of Virt, its transformation to MSC and obtaining the strict equivalence of MSC given on the relation is presented.

Keywords: The formal language, syntactic chart of Virt, the equivalent transformations, compiler program.

1. INTRODUCTION

Syntactic Virt charts represent visual, intuitively a clear, graphic way of a task of syntax of language, is used for documenting of programming languages [1, 2] and in projection of translators [3 - 7]. For creation of translators of the linear complexity [8, 9] the determined syntactic charts are used, thus the size of the translator depends on the size of the syntactic chart. Therefore, for the purpose of decrease of the size of the translator, it is expedient to use "compact" syntactic charts [10].

In article MSC, a way of transformation of the chart of Virt to MSC and its reductions to a form, "compact" in comparison with Virt's chart, are offered. The main difference of MSC from Virt's charts is that the MSC components can have more than one entrance.

2. MAIN PART

2.1 Definition of the syntactic chart of Virt

Let us determine the syntactic chart of Virt (SCV) by quintuple of D = (T, N, S, G, F), where T – finite set of terminals; N – finite set of not terminals; $S \in N$ – the initial not terminal; G = (V, E) – the focused count, where

$V = V_T \cup V_N \cup V_u \cup V_{ent} \cup V_{ex}$, where

V_{ent} – an entrance point set, $|V_{ent}| = |N|$;

V_T – a set of terminal tops;

V_N – a set of non-terminal tops;

V_u – finite set of clusters;

V_{ex} – an exit point set, $|V_{ex}| = |N|$;

$E = E_1 \cup E_2 \cup E_3 \cup E_4 \cup E_5$, *where*

$E_1 \subseteq \{(a, b) \mid a \in V_{ent}, b \in V_u\}$ – *a set of entrance arches;*

$E_2 \subseteq \{(a, b) \mid a \in V_u, b \in V_{ex}\}$ – *a set of entrance arches;*

$E_3 \subseteq \{(a, b) \mid a \in V_u, b \in V_T \cup V_N\}$ – *a set of the arches leaving clusters;*

$E_4 \subseteq \{(a, b) \mid a \in V_T \cup V_N, b \in V_u\}$ – *a set of the arches entering clusters;*

E5 $\subseteq$ {(a, b) | a $\in$ Vu,b $\in$ Vu}– a set of the ε-arches, connecting clusters;

$F: V_T \cup V_N \rightarrow T \cup N$ – display of a set of tops to a set of terminals and not terminals.

To each not terminal there corresponds the coherent component of the count. The component is called as the corresponding not terminal, has only one entry point and one point of an exit and finite set of tops of other types. An entry point and an exit on the chart of a component are not represented. The non-terminal top is represented by a rectangle in which the non-terminal symbol according to display F is entered. The terminal top is represented by a circle in which the terminal symbol according to display F is entered. The hub is represented on the chart by an aliphatic point. Any arch does not enter an entry point and there is a finite set of arches (entrance arches of a component). Clusters which entrance arches enter, are called initial. The point of an exit does not leave any arch and the finite set of arches (output arches of a component) enters. Clusters which leave output arches, are called closing. Each arch, except for entrance and output arches, can leave knot and enter terminal or non-terminal top or other knot, or to leave terminal or non-terminal top and to enter knot. Only one arch enters each terminal and non-terminal top and there is only one arch. On quantity of the arches which are entering clusters and leaving them restrictions are not present.

In a great number of all SCV can be allocated a class of the pseudo-determined SCV [8]. SCV belongs to a class of the pseudo-determined SCV (is pseudo-determined) if for it the following conditions are true:

– each component has only one initial knot;
– in it is not present ε-arches;
– any two arches leaving one knot enter the tops containing various symbols.

It is known [8, 10] that any SCV can be transformed in pseudo-determined therefore we will consider further only the pseudo-determined charts. In figure 1 the example of the pseudo-determined chart defining language is presented
$L = \{a\beta bb\beta a \mid \beta = a^*\}$ □ $\{b\beta aa\beta b \mid \beta = b^*\}$.

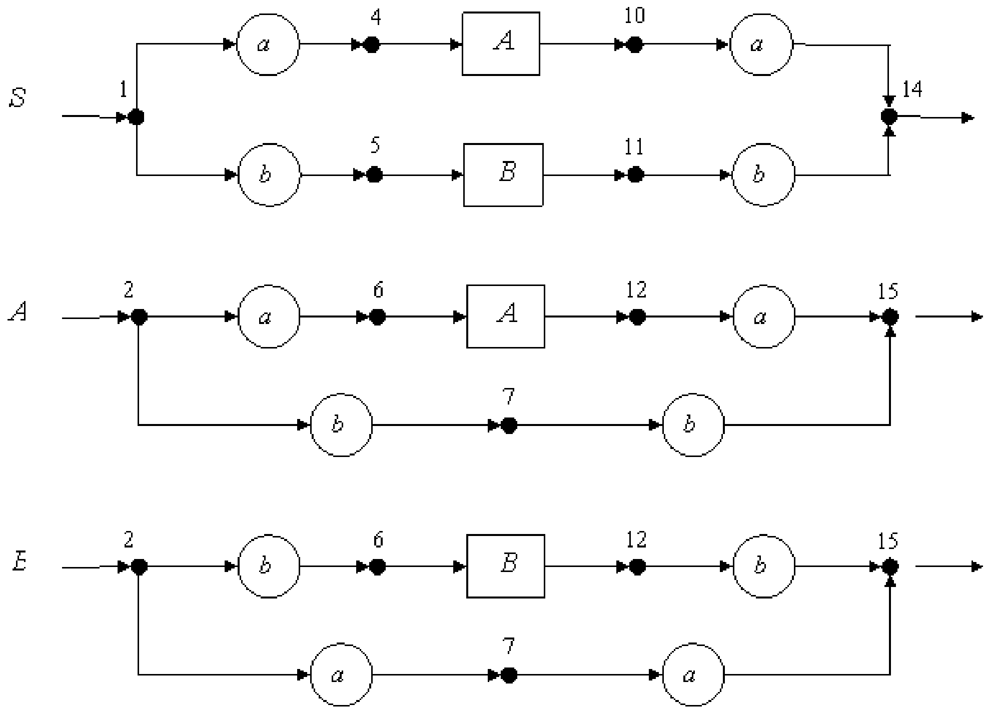

Fig. (1). – The pseudo-determined syntactic chart of Virt

Let us define a way of receiving a line-up of the language set by the pseudo-determined SCV.
Let us say that the line-up α, consisting of terminals and/or not terminals, connects a hub u the SCV components A with the closing u_k hub of this component if it can be received, "moving" to SC from a hub u to the u_k hub and writing out from the passing tops terminals and not terminals in initially empty line-up. A set of all line-ups connecting a hub u with closing clusters, we will designate L (u). Set of all line-ups connecting the initial u_0^A hub components A of the pseudo-determined SCV with closing, we will designate through L (And). It is apparent that $L(A) = L(u_0^A)$.

To receive a chain of the language set by the pseudo-determined SCV with the initial not terminal S, we will take the line-up belonging to L (S). If it contains not terminal A, we will replace it with a chain of a great number of L (And). If the received chain contains not terminal, similar actions is repeated. The chain received thus, free of not terminals, belongs to the language set by SCV.

Process of receiving one of line-ups of the language set by SCV in figure 1 can be presented as follows:

$S \Rightarrow aAa \Rightarrow aaAaa \Rightarrow aaaAaaa \Rightarrow aaaabbaaaa$

2.2 Definition of the modified syntactic chart

We will determine the modified syntactic chart (MSC) by the eight $R = (T, U, U', U'', u_0, G, F_T, F_U)$, where

T – finite set of terminals;

U – finite set of clusters;

U' – finite set of initial clusters, $U' \subseteq U$;

U'' – finite set of closing clusters, $U'' \subseteq U$;

u_0 – starting knot, $u_0 \in U'$;

$G = (V, E)$ – the focused equasion, where

$V = V_T \cup V_N \cup U$, where

V_T – a set of terminal tops;

V_N – a set of non-terminal tops;

$E = E_1 \cup E_2$, where

$E_1 \subseteq \{(a, b) \mid a \in U, b \in V_T \cup V_N\}$ – *a set of the arches leaving clusters and entering terminal or non-terminal tops;*

$E_2 \subseteq \{(a, b) \mid a \in V_T \cup V_N, b \in U\}$ – *a set of the arches leaving terminal or non-terminal tops and entering clusters;*

$F_T: V_T \rightarrow T$ – *display of a set of terminal tops in a set of terminals.*

$F_U: V_N \rightarrow U'$ – display of a set of non-terminal tops in a set of initial clusters.

The terminal top is represented on the chart by a circle in which the terminal according to the F_T display is entered. The non-terminal top is represented by a rectangle in which the knot from a set of initial clusters according to the F_U display is entered. The hub is represented by an aliphatic point which is noted by the corresponding number. Initial clusters are noted by the entering arrow, starting knot – the aliphatic entering vector, closing – the leaving arrow. The arch can leave terminal or non-terminal top and enter knot, or to leave knot and to enter terminal or non-terminal top. Only one arch enters each terminal and non-terminal top and there is only one arch. On quantity of the arches which are entering clusters and leaving clusters, restrictions are not present.

Let us define a way of receiving a line-up of the language set by MSC.

Let us say that the line-up α, consisting of terminals and/or initial clusters, knots the u MSC hub with closing u_k, if it can be received, "moving" to SC from a hub u to the u_k hub and writing out from the passing tops symbols (terminals or initial clusters) in initially empty line-up. A set of all line-ups connecting a hub u with closing clusters, we will designate $L(u)$. To receive a chain of the language set by MSC with the starting u_0, hub, we will take the line-up belonging to $L(u_0)$. If it contains some initial u_i, hub, we will replace it with a chain of a set of $L(u_i)$. If the received chain contains initial knot, similar actions is repeated. The chain received thus, free of initial clusters, belongs to the language set by MSC.

2.3 Transformation of the pseudo-determined syntactic chart of Virt to the modified syntactic chart

The pseudo-determined SCV can be transformed to MSC, having replaced not terminal which is written down in each non-terminal top with its initial knot and having defined initial knot of an initial component (corresponding to the initial not terminal) starting. In figure 2 MSC received from the pseudo-determined SCV (figure 1) is presented.

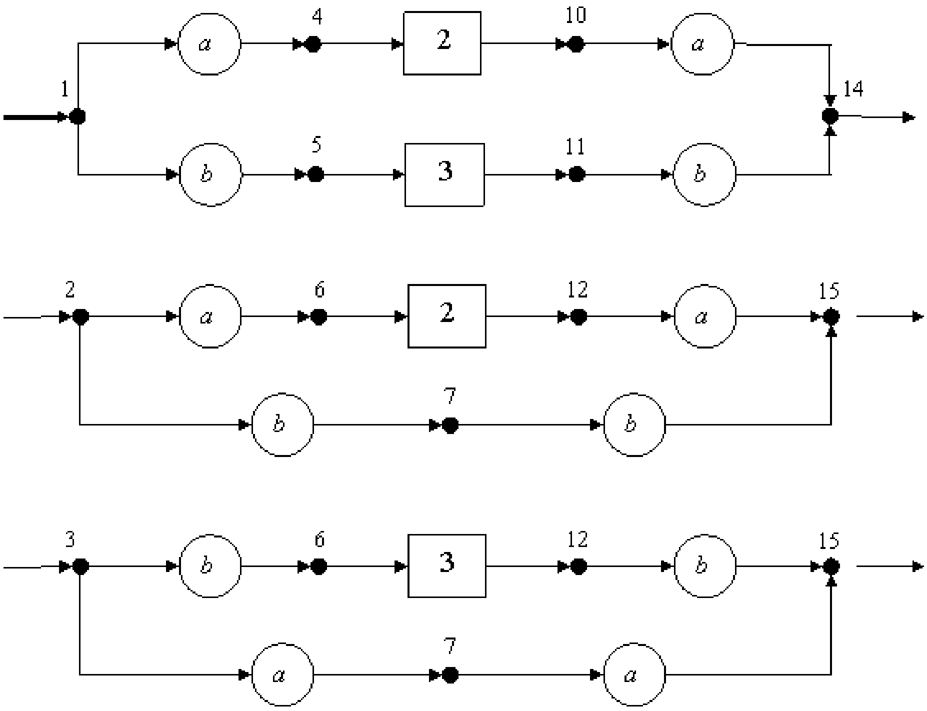

Fig. 2. The modified syntactic chart

Comparing ways of receiving line-ups of the language set by SCV and MSC it is possible to draw a conclusion that the language set by the pseudo-determined SCV is equal to the language set by MSC received from the pseudo-determined SCV on the way described above.

Process of receiving one of line-ups of the language set by MSC in figure 2 can be presented as follows:

1 □ *a*2*a* □ *aa*2*aa* □ *aaa*2*aaa* □ *aaaabbaaaa*

2.4 Equivalence relations on a set of clusters of the modified syntactic chart

Let MSC $R_i = (T, U, U', U'', u_i, G, F_T, F_U)$ and $R_j = (T, U, U', U'', u_j, G, F_T, F_U)$ be set. The charts R_i and R_j differ only with that in R_i hub is starting with u_i, and in R_j – with u_j. Let us designate through $L'(u_i)$ the language determined by MSC R_i, and through $L'(u_j)$ the language determined by MSC R_j. If $L'(u_i) = L'(u_j)$, then hubs u_i and u_j we call the equivalent (pair (u_i, u_j) belongs to an equivalence relation). Two MSC in which starting clusters are equivalent, also we will call the equivalent. The languages determined by the equivalent MSC – are equal.

Let us define more strong equivalence relation on a set of the MSC hubs and we will call it strict equivalence. Let us designate through the $L(u_i)$ set of all chains (consisting of terminals and initial clusters), connecting the ui hub with closing clusters, and through $L(u_j)$ – a set of all line-ups connecting the u_j hub with closing clusters. cluster pairs (u_i, u_j) belongs to the relation of strict equivalence, if $L(u_i) = L(u_j)$.

cluster pairs (u_i, u_j) belonging to the relation of strict equivalence belongs to an equivalence relation (the strictly equivalent clusters are the equivalent). cluster pairs (u_i, u_j) belonging to an equivalence relation can not belong to the relation of strict equivalence, i.e. the equivalent clusters can not be the strictly equivalent.

For two strictly equivalent clusters (u_i, u_j) the following conditions are true:

1) u_i and u_j hubs or both closing (u_i □U'' и u_j □ U''), or both not closing (u_i □U'' и u_j □ U'').
2) the quantity of the arches leaving the u_i hub is equal to quantity of the arches leaving the u_j hub.
3) the set of $M(u_i)$, consisting only of the symbols which are written down in tops to which there are arches from the u_i hub is equal to a set of $M(u_j)$, consisting only of the symbols which are written down in tops to which there are arches from the u_j hub.
4) if from the u_i (u_j) hub the arch goes to top with a symbol x, and from it the arch goes to the u_i' (u_j') hub, there is an arch which goes from the u_j (u_i) hub to top with a symbol x, and from it the arch goes to the u_j' (u_i') hub, and hubs u_i' and u_j' – are strictly equivalent.

The algorithm of definition of accessory of cluster pairs (u_i, u_j) to the relation of strict equivalence is based on check of a truth of the listed above conditions. Process of algorithm execution can be presented creation of a tree of check of strict equivalence in which tops couples of clusters register (in a root – cluster pairs checked for strict equivalence), and arches are noted by terminals or initial clusters. Tops of a tree under construction are defined as internal, boundary, duplicating, terminating or distinguishing. Boundary is the top which is a leaf and yet not processed by algorithm. After processing this top will become or internal, having sons, or duplicating, either terminating, or distinguishing. Duplicating is the top which is a leaf and containing such couple of states which is at least in one of internal tops. Terminating is a top which contains two identical knots. Distinguishing is a top which contains closing and not closing knot.

The algorithm of check of strict equivalence of two clusters contains three stages.

1. To construct top – a tree root, to write down in it couple of checked clusters. If for cluster pairs which are written down in a root one of the first three conditions of strict equivalence is false at least (see above), couple of checked clusters is not the strictly equivalent, the end of algorithm. If the clusters which are written down in a tree root closing and do not leave them an arch, couple of checked clusters is the strictly equivalent, the end of algorithm. Otherwise to consider a root boundary top and to execute clause 2.

2. So far there are boundary tops, to process them as construction as follows. Let in the processed top cluster pairs be written down (u_i, u_j). For each arch leaving the u_i hub to execute the following action. If the arch goes to top with a symbol x, and from it the arch goes to the u_i' hub, to find the arch leaving the u_j, hub which goes to top with a symbol x, and the u_j' hub, to which there is an arch from this top; to create top to which to carry out the arch noted by a symbol x, from the processed top; in the created top to write down cluster pairs (u_i', u_j').

If for cluster pairs which are written down in the created top one of the first three conditions of strict equivalence is false at least, to make this top distinguishing; couple of checked clusters is not the strictly equivalent, the end of algorithm.
If the created top contains such cluster pairs which is at least in one of internal tops, to make this top duplicating.
If the created top contains couple consisting of identical clusters, or steam of closing clusters which do not leave arches, to make this top terminating.
In other cases to consider the created top boundary.
To consider the processed boundary tops internal.

3. If in a set of tops of a tree is not present distinguishing, couple of checked clusters is the strictly equivalent, the end of algorithm.

If as a result of algorithm execution the tree in which there is no the distinguishing top is constructed, all couples of clusters which are written down in tree tops are the strictly equivalent. If as a result of algorithm execution the tree in which there is a distinguishing top is constructed, all couples of clusters which are written down in tops on the way from a root to the distinguishing top will not be the strictly equivalent.
Let us review examples of determination of the strict equivalence of some couples of MSC hubs represented in figure 2. In figure 3, and the tree of check of strict equivalence of hubs 1 and 2 is presented. In it there is a distinguishing top containing couple of hubs (5, 7) (for these clusters the third condition of strict equivalence is false) therefore hubs 1 and 2 are not the strictly equivalent.
In fig. 3, the tree of check of strict equivalence of hubs 4 and 6 is presented. The top containing hubs 14 and 15 – terminating (hubs 14 and 15 closing and do not leave them an arch). In this tree there is no the distinguishing top therefore hubs 4 and 6 are the strictly equivalent.

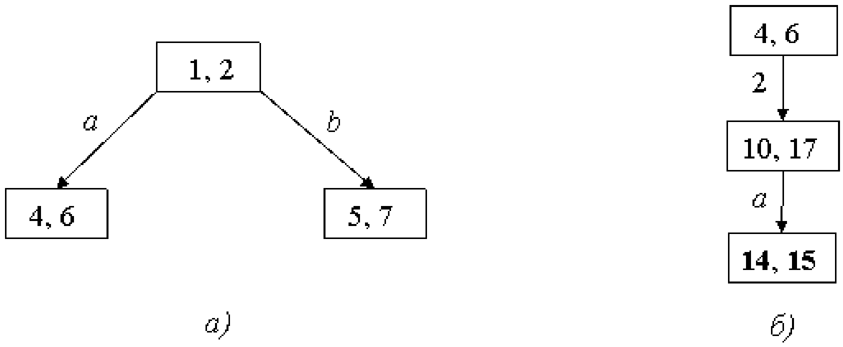

Fig. 3. Check of strict equivalence of clusters

2.5 The transformations keeping strict equivalence of the modified syntactic charts

The MSC transformation which for each initial knot does not change a set of the chains (consisting of terminals and initial clusters), connecting initial knot with closing, we will call the transformation keeping strict equivalence. It is natural that as a result of realization of the MSC transformation keeping strict equivalence MSC which defines the same language, as initial MSC will be received.

Let us consider three MSC transformations keeping strict equivalence.

First transformation.

Let in MSC $R = (T, U, U', U'', u_0, G, F_T, F_U)$ u_i and u_j hubs – initial and the strictly equivalent. Transformation is that the u_i hub is excluded from a set initial, and in all non-terminal tops containing the u_i hub, we replace it with the u_j hub. If one of the u_i or u_j hubs is starting, the u_j hub is considered starting. It is apparent that this transformation keeps strict equivalence.

Second transformation.

Let in MSC $R = (T, U, U', U'', u_0, G, F_T, F_U)$

$U' = \{u_0, u_1, \ldots u_f, \ldots, u_n\}$ – a set of initial clusters;

the u_i and u_j hubs – the strictly equivalent, the u_i hub – not initial;

any two initial knots are not the strictly equivalent.

Transformation is that the arches entering the ui hub will be redirected in the u_j hub.

Let us prove that this transformation keeps strict equivalence.

We choose from a set of initial hubs U' randomly the u_f hub. For descriptive reasons we will enter the following designations:

1) $L_1(u_f, u_i)$ – a set of the line-ups connecting the initial u_f hub with the u_i hub in initial MSC.
2) $L_1(u_i, U'')$ – a set of the line-ups connecting the u_i hub with closing hubs U'' in initial MSC.
3) $L_1(u_f, u_i, U'')$ – a set of the line-ups connecting the initial u_f hub with closing U'', corresponding to the paths passing through the u_i hub in initial MSC.

$L_1(u_f, u_i, U'') = L_1(u_f, u_i)\ L_1(u_i, U'')$ – a set of $L_1(u_f, u_i, U'')$ is equal to concatenation of sets $L_1(u_f, u_i)\ L_1(u_i, U'')$.

4) $L_1(u_{f,}, U'')$ – a set of the line-ups connecting the initial uf hub with closing U'', corresponding to the paths which are not passing through the u_i hub in initial MSC.

5) $L_1(u_f, U'')$ – a set of the line-ups connecting the initial uf hub with closing U'' in initial MSC.
$L_1(u_f, U'') = L_1(u_f, u_i, U'')\ \square\ L_1(u_{f,}, U'') = L_1(u_f, u_i)\ L_1(u_i, U'')\ \square\ L_1(u_{f,}, U'')$

6) $L_1(u_f, u_j)$ – a set of the line-ups connecting the initial u_f hub with the u_j hub in initial MSC.

7) $L_2(u_f, u_j)$ – a set of the line-ups connecting the initial u_f hub with the u_j hub in MSC received from initial as a result of realization of the considered transformation (in the transformed MSC).
$L_2(u_f, u_j) = L_1(u_f, u_j)\ \square\ L_1(u_f, u_i)$

8) $L_1(u_j, U'')$ – a set of the line-ups connecting the u_j hub with closing U'' in initial MSC.

9) $L_2(u_f, u_j, U'')$ – a set of the line-ups connecting the initial u_f hub with closing U'', corresponding to the paths passing through the uj hub in the transformed MSC.
$L_2(u_f, u_j, U'') = L_2(u_f, u_j)\ L_1(u_j, U'') = (L_1(u_f, u_j)\ \square\ L_1(u_f, u_i))\ L_1(u_j, U'') =$
$= L_1(u_f, u_j)\ L_1(u_j, U'')\ \square\ L_1(u_f, u_i)\ L_1(u_j, U'')$

10) $L_1(u_{f,}, U'')$ – a set of the line-ups connecting the initial u_f hub with closing U'', corresponding to the paths which are not passing through the u_j hub in initial MSC.

11) $L_2(u_f, U'')$ – a set of the line-ups connecting the initial u_f hub with closing U'' in the transformed MSC.
$L_2(u_f, U'') = L_2(u_f, u_j, U'')\ \square\ L_1(u_{f,}, U'') =$
$= L_1(u_f, u_j)\ L_1(u_j, U'')\ \square\ L_1(u_f, u_i)\ L_1(u_j, U'')\ \square\ L_1(u_{f,}, U'')$.

For the proof of that the considered transformation keeps strict equivalence, it is necessary to prove identity:
$L_1(u_f, U'') = L_2(u_f, U'')$, *thus*
$L_1(u_f, u_i)\ L_1(u_i, U'')\ \square\ L_1(u_{f,}, U'') =$
$= L_1(u_f, u_j)\ L_1(u_j, U'')\ \square\ L_1(u_f, u_i)\ L_1(u_j, U'')\ \square\ L_1(u_{f,}, U'')$.

We will carry out the proof by method of two inclusions.

1. Let us prove that $L_1(u_f, U'')\ \square\ L_2(u_f, U'')$. This inclusion is true, if
1. $L_1(u_f, u_i)\ L_1(u_i, U'')\ \square\ L_2(u_f, U'')$
2. $L_1(u_{f,}, U'')\ \square\ L_2(u_f, U'')$.

Truth of the first condition follows from this that $L_1(u_f, u_i)\ L_1(u_j, U'')$ is included in $L_2(u_f, U'')$ by definition of $L_2(u_f, U'')$ and equalities of $L_1(u_i, U'')$ and $L_1(u_j, U'')$ owing to strict equivalence of the u_i and u_j hubs.

Truth of the second condition follows from this that any chain of a set of $L_1(u_{f,}, U'')$ is included either in $L_1(u_f, u_j)$ $L_1(u_j, U'')$, or in $L_1(u_{f,}, U'')$, i.e. it corresponds or to the way passing through the u_j, hub, or to the way which is not passing through the hub u_j.

2. Let us prove that $L_2(u_f, U'')$ □ $L_1(u_f, U'')$. This inclusion is true, if

1. $L_1(u_f, u_j)$ $L_1(u_j, U'')$ □ $L_1(u_f, U'')$
2. $L_1(u_f, u_i)$ $L_1(u_j, U'')$ □ $L_1(u_f, U'')$
3. $L_1(u_{f,}, U'')$ □ $L_1(u_f, U'')$

Truth of the first condition follows from this that any chain of a set of $L_1(u_f, u_j)$ $L_1(u_j, U'')$ included either in $L_1(u_f, u_i)$ $L_1(u_i, U'')$, or in $L_1(u_{f,}, U'')$, i.e. it corresponds or to the way passing through the u_i hub, or to the way which is not passing through the u_i hub.
Truth of the second condition follows from this that $L_1(u_f, u_i)$ $L_1(u_i, U'')$ is included in $L_1(u_f, U'')$ by definition $L_1(u_f, U'')$ and equality $L_1(u_i, U'')$ and $L_1(u_j, U'')$ owing to strict equivalence of the u_i and u_j.
Truth of the third condition follows from this that any chain of a set of $L_1(u_{f,}, U'')$ included either in $L_1(u_f, u_i)$ $L_1(u_i, U'')$, or in $L_1(u_{f,}, U'')$, i.e. it corresponds or to the way passing through the u_i hub, or to the way which is not passing through the u_i hub.
Both inclusions are proved, therefore the considered transformation keeps strict equivalence.

Third transformation.
This transformation is bound to an exception of tops of MSC (clusters, terminal and non-terminal tops), unattainable of initial clusters, i.e. tops to which there is no way from any initial knot. Removal of such tops together with the leaving arches does not change the language determined by MSC.
As a result of realization of the first two transformations keeping strict equivalence of MSC new MSC in which any arch does not enter some clusters turns out. Such clusters are unattainable from initial and therefore they can be removed. Let u – knot which any arch does not enter; e_i – an arch which leaves a hub u and enters terminal or non-terminal top of v; e_j – an arch which leaves v top. Together with a hub u arches of e_i and e_j and top of v are removed.
If as a result of realization of the actions described above MSC with clusters which arches do not enter again turns out, to repeat operation.

2.6 The modified syntactic charts provided on the relation of strict equivalence
Let us consider that in initial MSC there are no the clusters unattainable from initial, and there are no clusters from which closing clusters are unattainable.
Carrying out the transformations described in the section 5 over initial MSC while it is possible, MSC in which there is no couple of the strictly equivalent clusters, the clusters unattainable from initial, and clusters from which the closing are unattainable will be received. We will call such MSC the strict equivalence given on the relation, and process of receiving the given MSC – reduction of MSC. If initial MSC is not brought, the given MSC will contain less (in comparison with initial) number of knots, arches, terminal and non-terminal tops, and, perhaps, and smaller quantity of coherent components. The quantity of components decreases in case in initial MSC

there is such couple of the strictly equivalent u_i and u_j hubs that the u_i hub belongs to one coherent component, and the u_j hub – another. Then components which possess the u_i and u_j hubs, as a result of realization of transformations, "communicate" in one in which the set of initial clusters will be equal to union of sets of initial clusters of these components.

Process of receiving the given MSC of MSC represented in figure 2 is presented in figure 4. Initial MSC has rather prime structure allowing to define with little effort couples of the strictly equivalent clusters. Hubs 4 and 6 the strictly equivalent. Carrying out transformations, we will receive MSC presented in figure 4, and. In this MSC hubs 5 and 8 the strictly equivalent. Carrying out transformations, we will receive MSC presented in figure 4. In the received MSC hubs 9 and 12 the strictly equivalent. Carrying out transformations, we will receive MSC presented in figure 4, century. In it hubs 13 and 7 are the strictly equivalent. Carrying out transformations, we will receive MSC presented in figure 4. In this MSC there is no couple of the strictly equivalent clusters and clusters unattainable from initial therefore it is brought. In more visual form the given MSC is presented in figure 5. In comparison with initial, it contains twice less clusters, one coherent component and three initial knots.

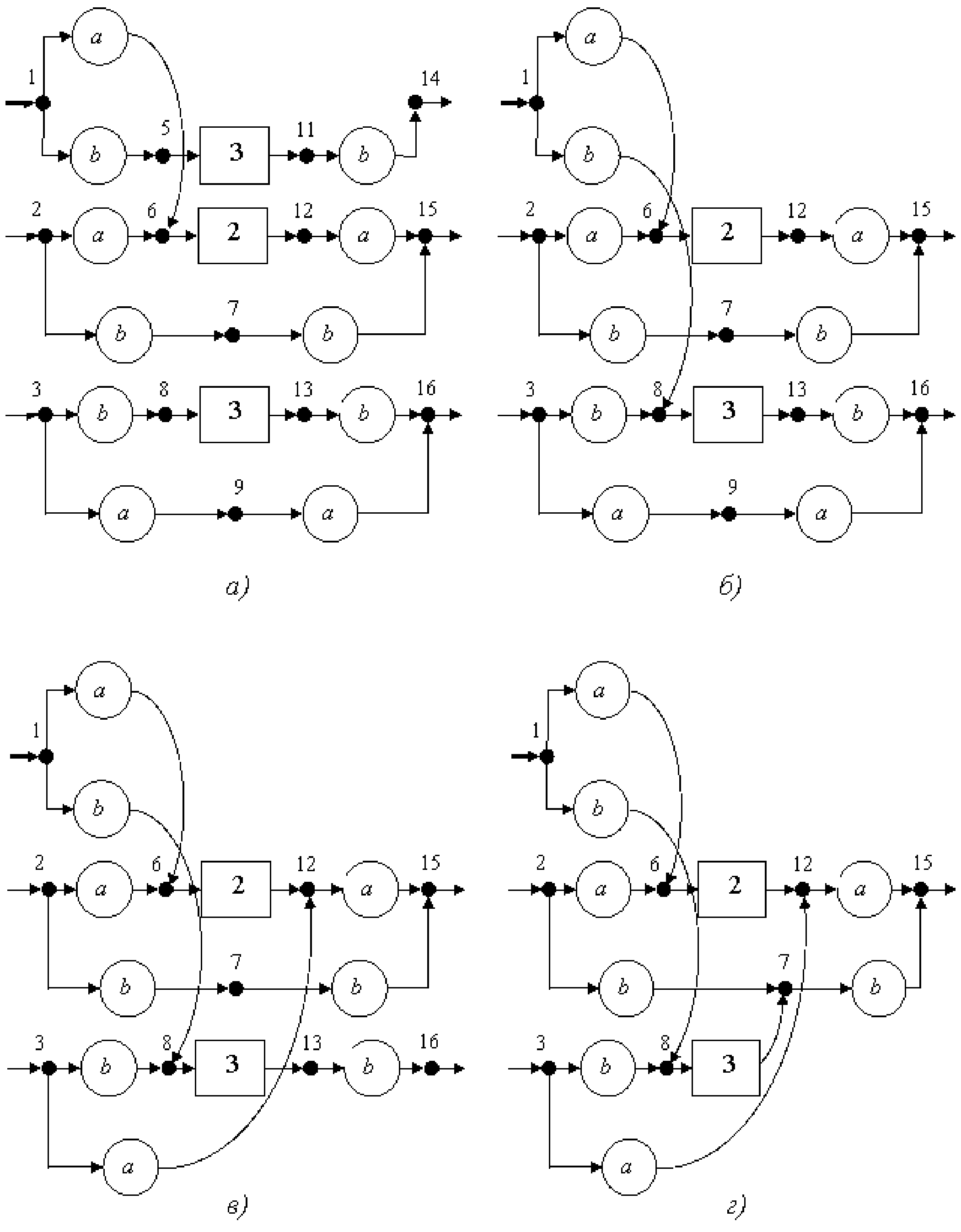

Fig. 4. Process of receiving the given MSC

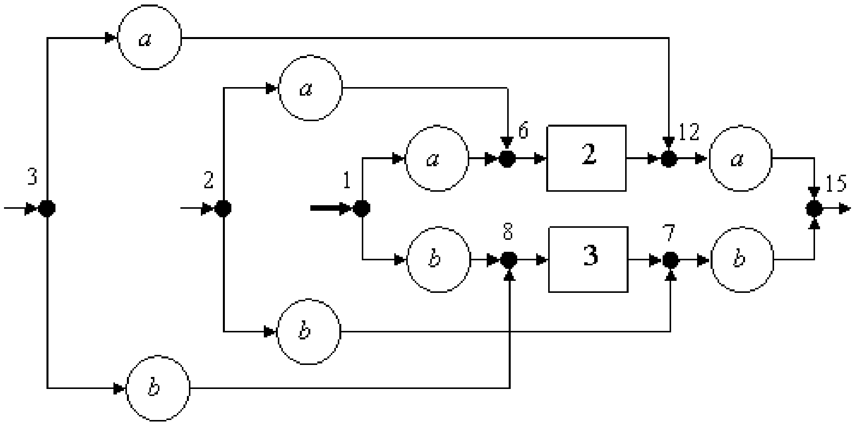

Fig. 5. The given MSC

3. SUMMARY

The modified syntactic charts provided on the relation of strict equivalence is a compact way of representation of the formal language and can be used for creation of translators, efficient on memory. Application of other types of equivalence for reduction of the size of the modified syntactic charts can be development of this work.

4. CONCLUSION

On the basis of the above it is possible to draw a conclusion that representation of the formal language the modified syntactic chart provided on the relation of strict equivalence is more compact, than its representation by the pseudo-determined syntactic chart of Virt.

5. ACKNOWLEDGEMENTS

Work was prepared with assistance of the Russian fund of basic researches (grant No. 12-07-00493/14).

REFERENCES

1. Jensen K., Wirth N., 1975. Pascal User Manual and Report. Springer-Verlag, New York, pp: 167.
2. Jensen K., Wirth N., 1996. Pascal Standard Iso. Jackson Libri, pp: 290.
3. Legalov A. I. Bases of development of translators. Date of the address 22.04.2015 http://www.softcraft.ru/translat/
4. Legalov A.I., Shvets D. A., Legalov I.A., 2007. The formal languages and translators. Krasnoyarsk. The Siberian federal university, with: 213.

5. Karpov Yu.G., 2005. Theory and technology of programming. Bases of creation of translators. SPb. BHV-St. Petersburg, with: 272.
6. Sverdlov S.Z., 2007. Programming languages and methods of broadcasting. SPb. St. Petersburg, with: 638.
7. Martynenko B. K., 2014. Syntactic charts of N. Virt and columns schemes in Syntax-technology. Computer tools in education, 2: 3 – 19.
8. Ryazanov YU.D., Sevalneva M. N., 2013. Analysis of syntactic charts and synthesis of programs recognizers of the linear complexity. Scientific sheets of BELGU. It is gray. History. Political science. Economy. Informatics, 8(151): 128–136.
9. Poles V.M., Ryazanov Yu.D., 2013. Algorithm of creation of nonrecursive programs recognizers of the linear complexity for the determined syntactic charts. The BGTU bulletin of V. G. Shukhov, 6: 194 – 199.
10. Ryazanov YU.D., 2015. Transformation of nondeterministic syntactic charts to the determined. Bulletin of VSU, series: systems analysis and informational technologies, 1: 139 — 147.
11. Ryazanov YU.D., 2015. Decrease of quantity of components of the determined syntactic chart on the basis of the strong equivalence. Intelligence systems and technologies, 1(87): 94 – 100.