Rytter for CFPQ

Semyon Grigorev
Saint Petersburg State University
St. Petersburg, Russia
semen.grigorev@jetbrains.com

Ekaterina Shemetova
Saint Petersburg State University
St. Petersburg, Russia
nozhkin.ii@gmail.com

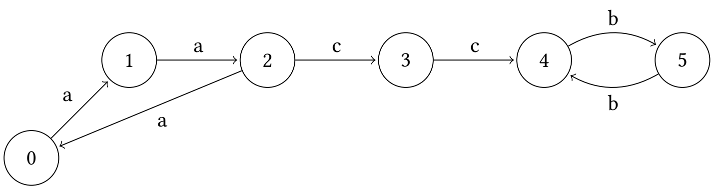


Figure 1: The input graph

ACM Reference Format:

Semyon Grigorev and Ekaterina Shemetova. 2018. Rytter for CFPQ. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

We provide an idea of two steps reduction of CFPQs to Boolean matrix multiplication. First step is reduction of arbitrary CFPQ to Dyck query. Second step is adoptation Rytter's results from [2] for graph.

Finally we discuss "fully algebraic" view of CFPQ complexity.

2 FROM ARBITRARY CFPQ TO DYCK QUERY

This reduction is inspired by the construction described in [1].

Consider a context-free grammar $\mathcal{G} = (\Sigma, N, P, S)$ in BNF where Σ is a terminal alphabet, N is a nonterminal alphabet, P is a set of productions, $S \in N$ is a start nonterminal. Also we denote a directed labeled graph by $G = (V, E, L)$ where $E \subseteq V \times L \times V$ and $L \subseteq \Sigma$.

We should construct new input graph G' and new grammar $\mathcal{G}'$ such that $\mathcal{G}'$ specifies a Dyck language and there is a simple mapping from CFPQ($\mathcal{G}', G'$) to CFPQ($\mathcal{G}, G$). Step-by-step example with description is provided below.

Let the input grammar is

$$S \to a\ S\ b \mid a\ C\ b$$
$$C \to c \mid C\ c$$

The input graph is presented in fig. **??**

(1) Let $\Sigma_{()} = \{t_(, t_) | t \in \Sigma\}$.
(2) Let $N_{()} = \{N_(, N_) | N \in N\}$.

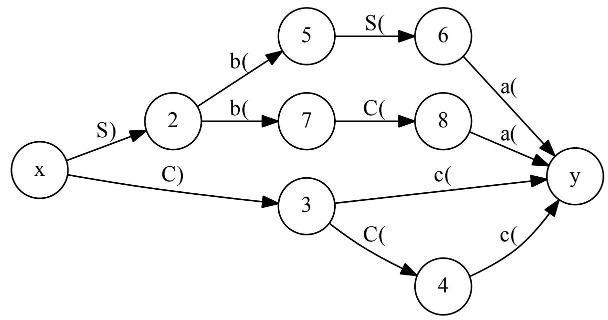


Figure 2: The input graph

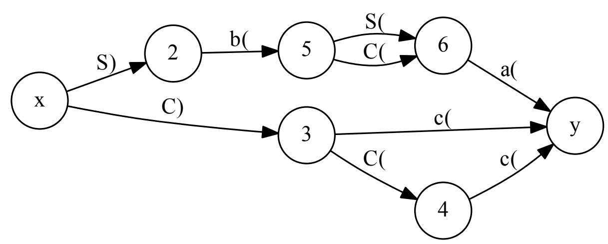


Figure 3: The input graph

(3) Let $M_{\mathcal{G}} = (V_{\mathcal{G}}, E_{\mathcal{G}}, L_{\mathcal{G}})$ is a directed labeled graph, where $L_{\mathcal{G}} \subseteq (\Sigma_{()} \cup N_{()})$. This graph is created the same manner as described in [1] but we do not require the grammar be in CNF. Let $x \in V_{\mathcal{G}}$ and $y \in V_{\mathcal{G}}$ is "start" and "final" vertices respectively. This graph may be treated as a finite automaton, so it can be minimized and we can compute an ε-closure if the input grammar contains ε productions. The graph $M_{\mathcal{G}}$ for our example is:
The minimized graph:
(4) For each $v \in V$ create $M^v_{\mathcal{G}}$: unique instance of $M_{\mathcal{G}}$.
(5) New graph $G^{'}$ is a graph G where each label t is replaced with $t^i_)$ and some additional edges are created:
   - Add an edge $(v', S_(, v)$ for each $v \in V$.
   - And the respective $M^v_{\mathcal{G}}$ for each $v \in V$:
     - reattach all edges outgoing from x^v ("start" vertex of $M^v_{\mathcal{G}}$) to v;
     - reattach all edges incoming to y^v ("final" vertex of $M^v_{\mathcal{G}}$) to v.

   New input graph is ready:

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM...$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

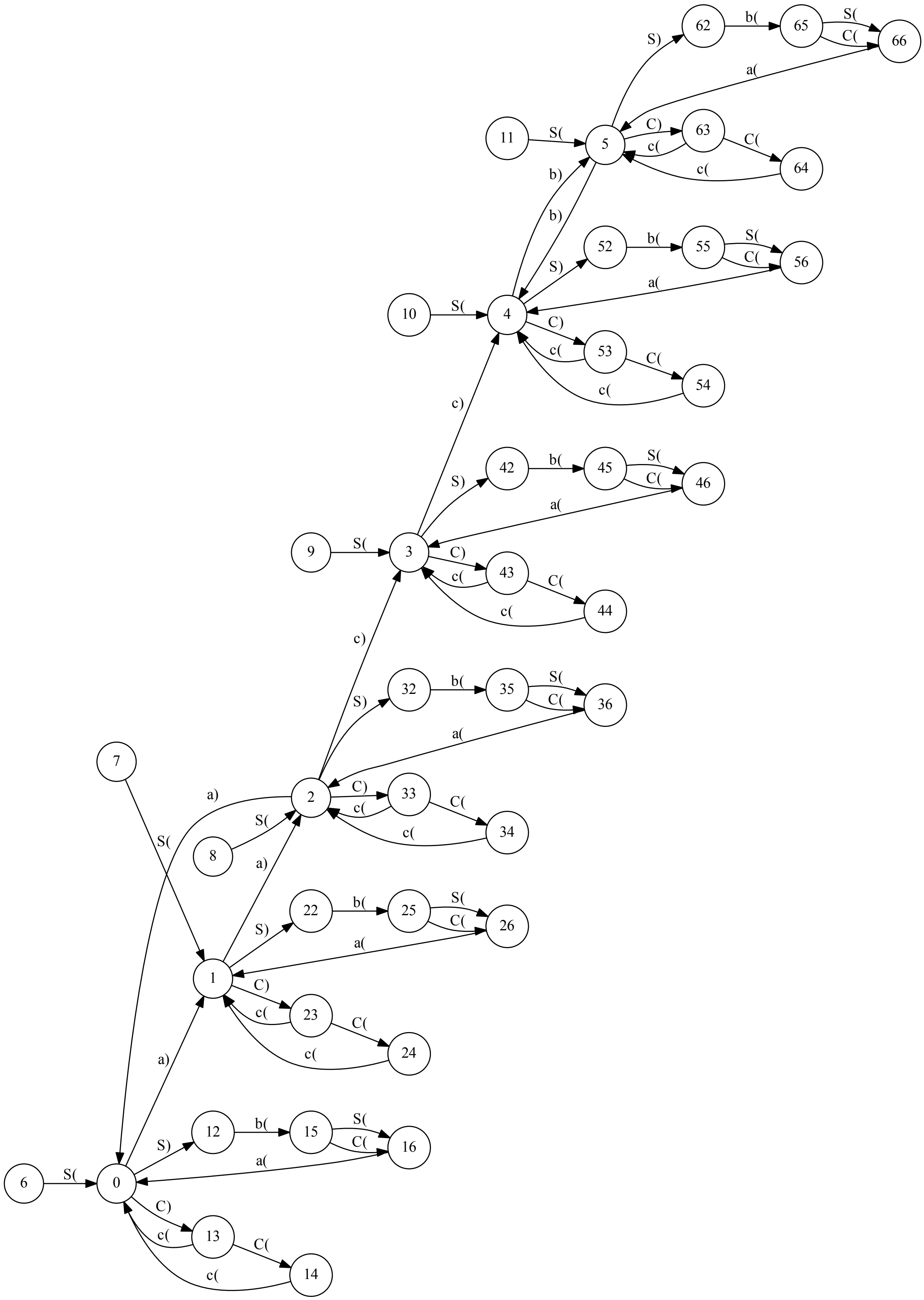


Figure 4: The same generation query (Query 2) in Meerkat

(6) New grammar $\mathcal{G}' = (\Sigma', N', P', S')$ where $\Sigma' = \Sigma_{()} \cup N_{()}$, $N' = \{S'\}$, $P' = \{S' \to b_(\ S' \ b_); S' \to b_(\ b_) \mid b_(, b_) \in \Sigma'\} \cup \{S' \to S' \ S'\}$ is a set of productions, $S' \in N'$ is a start nonterminal.

Now, if CFPQ($\mathcal{G}', G'$) contains a pair (u_0', v') such that $e = (u_0', S_(, u_1') \in E'$ is an extension edge (step 5, first subitem), then $(u_1', v') \in$ CFPQ($\mathcal{G}, G$). In our example, we can find the following path: $7 \xrightarrow{S_(} 1 \xrightarrow{S_)} 22 \xrightarrow{b_(} 25 \xrightarrow{C_(} 26 \xrightarrow{a_(} 1 \xrightarrow{a_)} 2 \xrightarrow{C_)} 33 \xrightarrow{C_(} 34 \xrightarrow{c_(} 2 \xrightarrow{c_)} 3 \xrightarrow{C_)} 43 \xrightarrow{c_(} 3 \xrightarrow{c_)} 4 \xrightarrow{b_)} 5$. Edge $7 \xrightarrow{S_(} 1$ is the extension, so (1,5) should be in CFPQ($\mathcal{G}, G$) and it is true.

3 RYTTER ALGORITHM FOR GRAPH INPUT

Main idea is to adopt algorithm from [2] for CFPQ. It should be possible to perform adoptation for arbitrary CFPQ, but we are interested in case of Dyck queryes.

We introduce an example and try to explain key steps. As far as example for graph and query introduced in the previous section is too big, we use another input data.

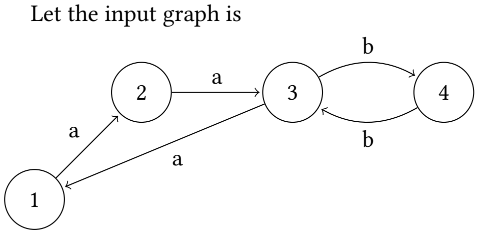
Let the input grammar is

$$S \to a \ S \ b$$
$$S \to a \ b$$

The input grammar in CNF is

$$S \to A \ S_1$$
$$S_1 \to S \ B$$
$$S \to A \ B$$
$$A \to a$$
$$B \to b$$

Let the input graph is



The *IMPLIED* relation:

Grid:

We should introduce the *id* implication such that for every $A \in$ IMPLIED

- $id \times A = A \times id$

In order to compute transitive closure in logarithmic time we add self-loop with weight $\{id\}$ to each vertex.

Note that our graph is a Cartezian product of tho graph H and V with respective matrices.

$H =$

$$\begin{pmatrix} \{id\} & \varnothing & \varnothing & \varnothing \\ \varnothing & \{id\} & \varnothing & \varnothing \\ \varnothing & \varnothing & \{id\} & \{(A,S);(S,S_1)\} \\ \varnothing & \varnothing & \{(A,S);(S,S_1)\} & \{id\} \end{pmatrix}$$

$V =$

$$\begin{pmatrix} \{id\} & \varnothing & \varnothing & \varnothing \\ \varnothing & \{id\} & \{(B,S);(S_1,S)\} & \varnothing \\ \varnothing & \varnothing & \{id\} & \{(B,S);(S_1,S)\} \\ \varnothing & \{(B,S);(S_1,S)\} & \varnothing & \{id\} \end{pmatrix}$$

Matrix of $G = V \otimes I + I \otimes H$ where I is identity matrix of size $n \times n$ and $\otimes$ is a Kronecker product.

One step is APSP (or transitive closure) of G. It can be computed as $(V \otimes I + I \otimes H)^{(n^2)}$. Now we should check validity of nonterminals. It can be don by multiplication of vector x and M. $x * (V^{(n^2)} \otimes I + V^{(n^2)} \otimes H^{(n^2)} + I \otimes H^{(n^2)}) = x * V^{(n^2)} \otimes I + x * V^{(n^2)} \otimes H^{(n^2)} + x * I \otimes H^{(n^2)}$.

$X =$

$$\begin{pmatrix} \varnothing & \varnothing & \{(\bot,B)\} & \varnothing \\ \{(\bot,A)\} & \varnothing & \varnothing & \{(\bot,B)\} \\ \varnothing & \varnothing & \{(\bot,A)\} & \varnothing \\ \varnothing & \{(\bot,A)\} & \varnothing & \varnothing \end{pmatrix}$$

$X^T =$

$$\begin{pmatrix} \varnothing & \{(\bot,A)\} & \varnothing & \varnothing \\ \varnothing & \varnothing & \varnothing & \{(\bot,A)\} \\ \{(\bot,B)\} & \varnothing & \{(\bot,A)\} & \varnothing \\ \varnothing & \{(\bot,B)\} & \varnothing & \varnothing \end{pmatrix}$$

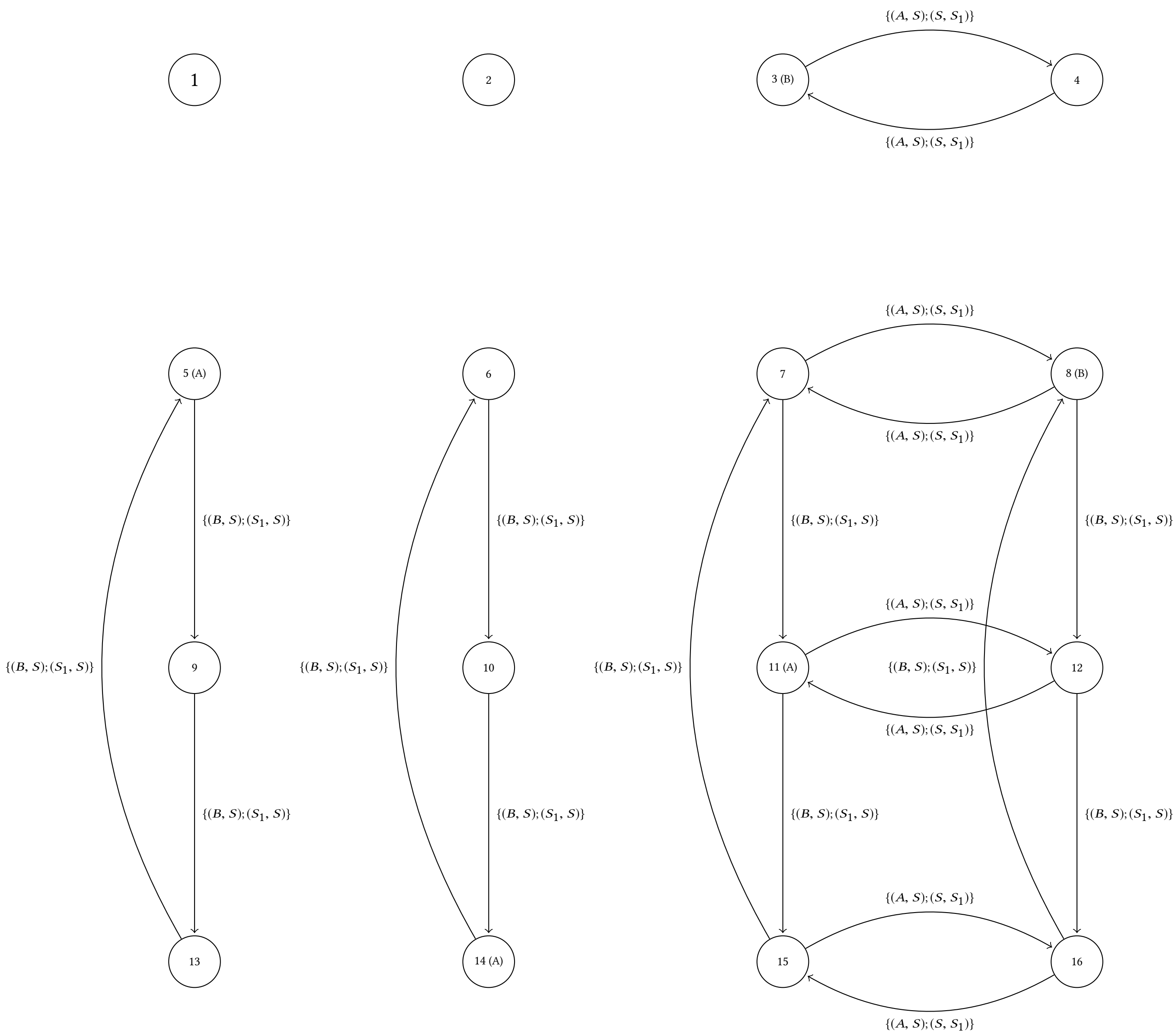
4 ALGEBRAIC VIEW

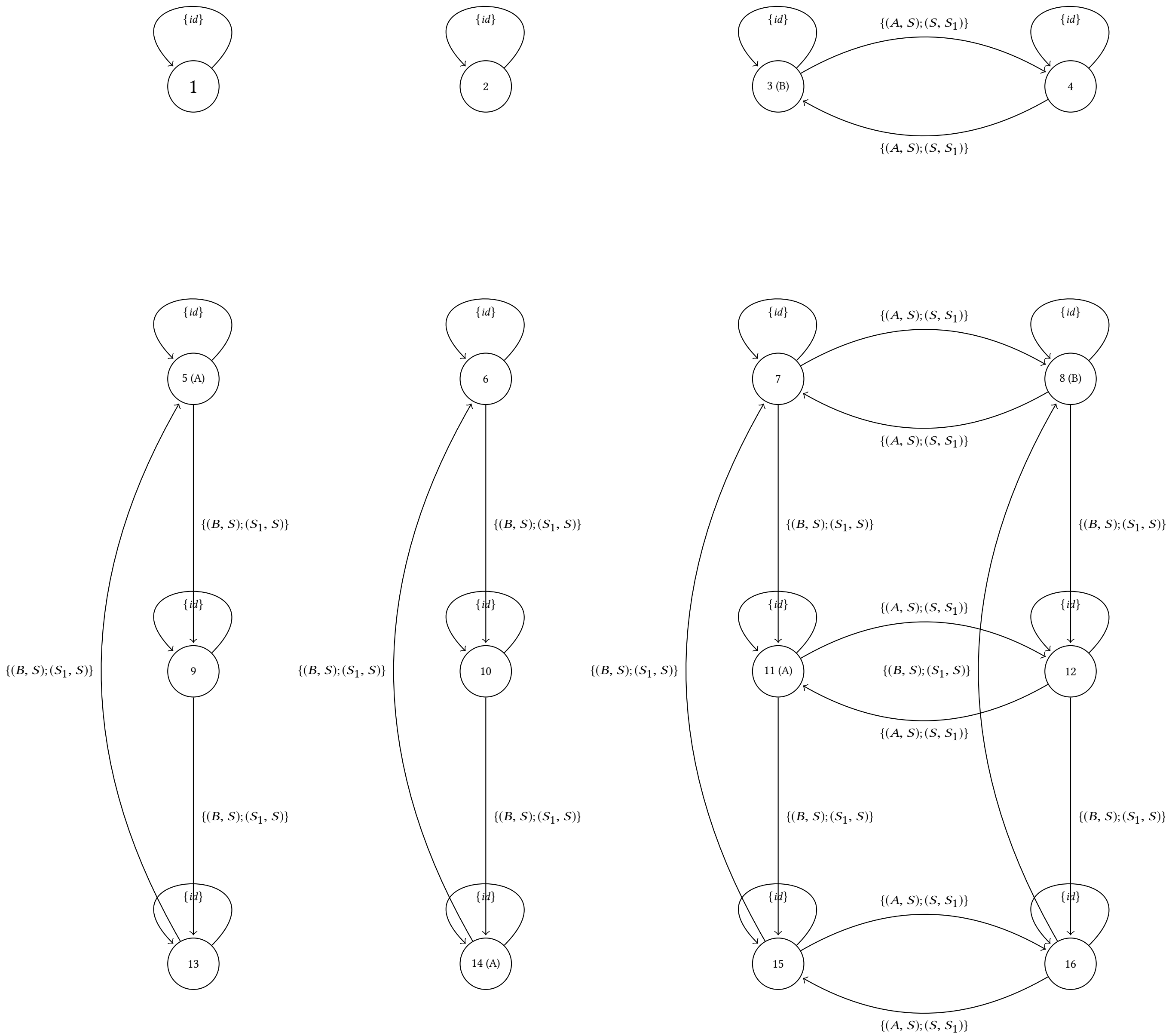
REFERENCES

[1] Krishnendu Chatterjee, Bhavya Choudhary, and Andreas Pavlogiannis. 2017. Optimal Dyck Reachability for Data-dependence and Alias Analysis. *Proc. ACM Program. Lang.* 2, POPL, Article 30 (Dec. 2017), 30 pages. https://doi.org/10.1145/3158118

[2] Wojciech Rytter. 1995. Context-free recognition via shortest paths computation: a version of Valiant's algorithm. *Theoretical Computer Science* 143, 2 (1995), 343–352.

$$
\begin{array}{cccc}
(B,2,3) \Rightarrow (S,1,3) & (B,2,4) \Rightarrow (S,1,4) & (B,2,2) \Rightarrow (S,1,2) & (B,2,1) \Rightarrow (S,1,1) \\
(B,3,4) \Rightarrow (S,2,4) & (B,3,3) \Rightarrow (S,2,3) & (B,3,2) \Rightarrow (S,2,2) & (B,3,1) \Rightarrow (S,2,1) \\
(B,1,2) \Rightarrow (S,3,2) & (B,1,3) \Rightarrow (S,3,3) & (B,1,4) \Rightarrow (S,3,4) & (B,1,1) \Rightarrow (S,3,1) \\
(S_1,2,3) \Rightarrow (S,1,3) & (S_1,2,4) \Rightarrow (S,1,4) & (S_1,2,2) \Rightarrow (S,1,2) & (S_1,2,1) \Rightarrow (S,1,1) \\
(S_1,3,4) \Rightarrow (S,2,4) & (S_1,3,3) \Rightarrow (S,2,3) & (S_1,3,2) \Rightarrow (S,2,2) & (S_1,3,1) \Rightarrow (S,2,1) \\
(S_1,1,2) \Rightarrow (S,3,2) & (S_1,1,3) \Rightarrow (S,3,3) & (S_1,1,4) \Rightarrow (S,3,4) & (S_1,1,1) \Rightarrow (S,3,1) \\
(A,2,3) \Rightarrow (S,2,4) & (A,1,3) \Rightarrow (S,1,4) & (A,3,3) \Rightarrow (S,3,4) & (A,4,3) \Rightarrow (S,4,4) \\
(A,3,4) \Rightarrow (S,3,3) & (A,4,4) \Rightarrow (S,4,3) & (A,2,4) \Rightarrow (S,2,3) & (A,1,4) \Rightarrow (S,1,3) \\
(S,2,3) \Rightarrow (S_1,2,4) & (S,1,3) \Rightarrow (S_1,1,4) & (S,3,3) \Rightarrow (S_1,3,4) & (S,4,3) \Rightarrow (S_1,4,4) \\
(S,3,4) \Rightarrow (S_1,3,3) & (S,4,4) \Rightarrow (S_1,4,3) & (S,2,4) \Rightarrow (S_1,2,3) & (S,1,4) \Rightarrow (S_1,1,3)
\end{array}
$$




{id}
1
2
3 (B)
4
$\{(A, S); (S, S_1)\}$
5 (A)
6
7
8 (B)
9
10
11 (A)
12
13
14 (A)
15
16
$\{(B, S); (S_1, S)\}$