# From grammar to shared packed parse forest

Semyon Grigorev
Saint Petersburg State University
7/9 Universitetskaya nab.
St. Petersburg, 199034 Russia
semen.grigorev@jetbrains.com

It is possible to build graph structured representation of result which can be interesting for several reasons.

- More user friendly query result represenation.
- Useful for query result explorationa and investigation.
- Useful for query debugging.

Example from your article [2] will be used for explanation. Let we intrioduce new notation for nonterminals and terminals for simplification.

$$\begin{array}{ll} q_1 = q[A,B] & q_5 = q[C,E] \\ q_2 = q[A,C] & q_6 = q[A,D] \\ q_3 = q[D,E] & q_7 = q[A,E] \\ q_4 = q[B,D] & q_8 = q[B,E] \end{array}$$

$$\begin{array}{ll} T_1 = friendOf_{A,B} & T_4 = friendOf_{B,D} \\ T_2 = friendOf_{A,C} & T_5 = friendOf_{C,E} \\ T_3 = friendOf_{D,E} & \end{array}$$

Now grammar from [2](pages 6-7) can be represented as below. Let it be named $G_1$

$$\begin{array}{ll} q_1 \to T_1 & q_6 \to q_1\ q_4 \\ q_2 \to T_2 & q_7 \to q_2\ q_5 \\ q_3 \to T_3 & q_7 \to q_1\ q_8 \\ q_4 \to T_4 & q_7 \to q_6\ q_3 \\ & q_8 \to q_4\ q_3 \end{array}$$

For each context-free grammar we can create regular tree grammar [1](section 2.4). Moreover if $G$ is a context-free word grammar, then the set of derivation trees of $L(G)$ is a regular tree language. By using algorithm presented in [1](section 2.4) we can build regular tree grammar $G_2$ for grammar $G_1$:

$$\begin{array}{ll} q_1 \to q_1(T_1) & q_6 \to q_6(q_1, q_4) \\ q_2 \to q_2(T_2) & q_7 \to q_7(q_2, q_5) \\ q_3 \to q_3(T_3) & q_7 \to q_7(q_1, q_8) \\ q_4 \to q_4(T_4) & q_7 \to q_7(q_6, q_3) \\ & q_8 \to q_8(q_4, q_3) \end{array}$$

Reduced graph. It is SPPF.

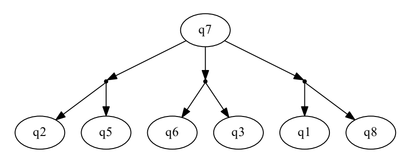


Figure 1: Input graph $M$

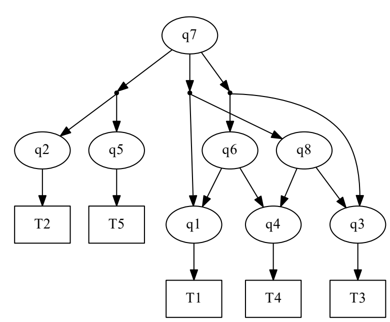


Figure 2: Input graph $M$

Let return information about edges back.

It is possible to build SPPF with our tool without to CNF transformation.

Query grammar.

Input graph.

Query result

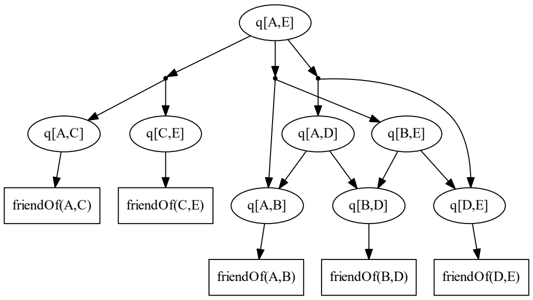


Figure 3: SPPF with information about paths

```
0: s = L s R
1: s = middle
2: middle = L R
```

Figure 4: Grammar $G_1$ for language $L = \{L^n R^n; n \geq 1\}$

# 1. REFERENCES

[1] Comon, Hubert, et al. "Tree automata techniques and applications." (2007).

[2] Hellings, Jelle. "Querying for Paths in Graphs using Context-Free Path Queries." arXiv preprint arXiv:1502.02242 (2015).

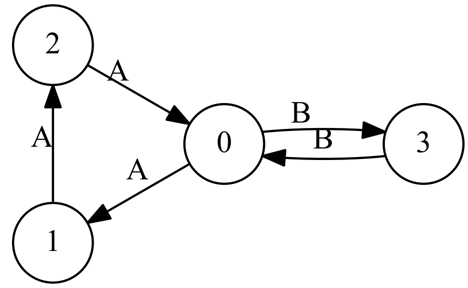


Figure 5: SPPF with information about paths

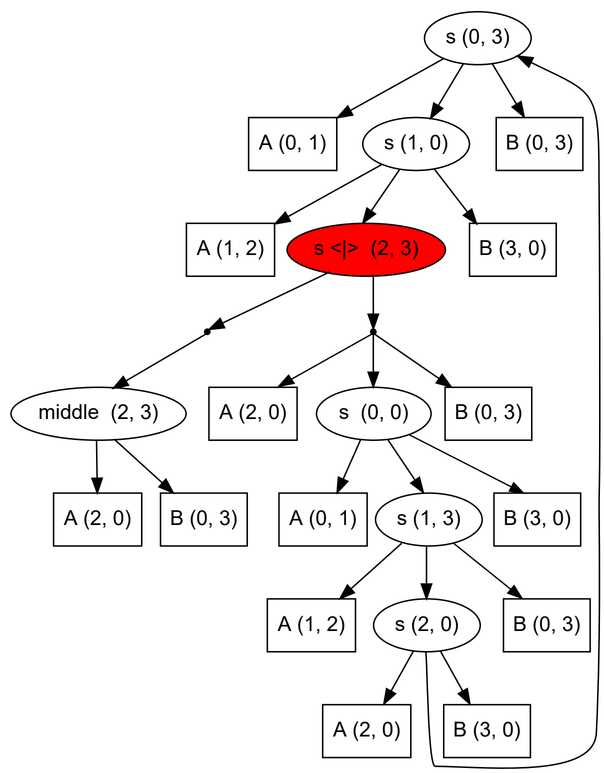


Figure 6: SPPF with information about paths