Transition-Based Coding and Formal Language Theory for Ordered Digraphs

–

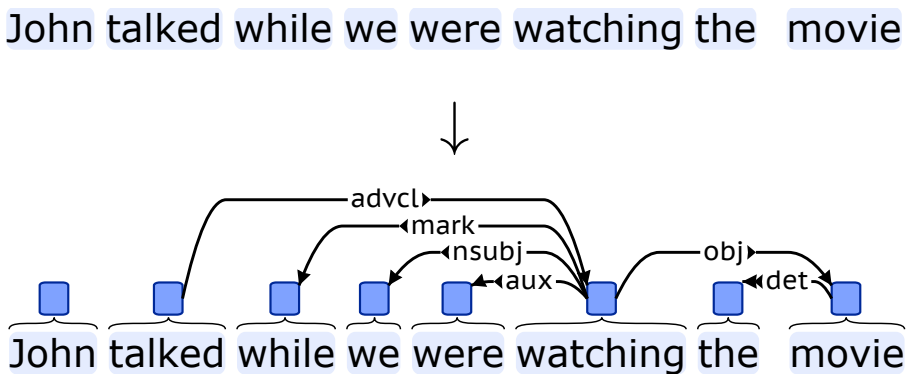
Embedding Arbitrary Graphs to the String Space
to Support Succinct Representation of Infinite Sets of Graphs

Anssi Yli-Jyrä
University of Helsinki, Finland

(1) FSMNLP, September 23–25, 2019, Dresden
(2) LT Research Seminar, October 24, 2019, Helsinki
(3) Grupo LYS, November 12, 2019, A Coruña

(4) NLP Reading Club, December 16, 2019, Jerusalem

Parsing produces Graphs



Transition-Based Parsing

$\mathbf{x} = x_1 \ldots x_m \in \Sigma^*$: input string,

C : set of configurations, $C_I \subseteq C$: set of intial configurations

$c_0 \in C_I$: initial configuration

Transition-Based Parsing

$\mathbf{x} = x_1 ... x_m \in \Sigma^*$: input string,

C : set of configurations, $C_I \subseteq C$: set of intial configurations

$c_0 \in C_I$: initial configuration

$a_1, a_2, ... \in A$: actions, $\quad c_1, c_2, ... \in C$

$c_0 \overset{a_1 ... a_n}{\longrightarrow} c_n, c_0 \in C_I, c_n \in C_{F_\mathbf{x}}$: valid action sequences V

Transition-Based Parsing

$\mathbf{x} = x_1 ... x_m \in \Sigma^*$: input string,

C : set of configurations, $C_I \subseteq C$: set of intial configurations

$c_0 \in C_I$: initial configuration

$a_1, a_2, ... \in A$: actions, $\quad c_1, c_2, ... \in C$

$c_0 \xrightarrow{a_1 ... a_n} c_n, c_0 \in C_I, c_n \in C_{F_\mathbf{x}}$: valid action sequences V

$P(c_0 \xrightarrow{a_1} c_1 \mid \mathbf{x}, c_0)$

Transition-Based Parsing

$\mathbf{x} = x_1 ... x_m \in \Sigma^*$: input string,

C : set of configurations, $C_I \subseteq C$: set of intial configurations

$c_0 \in C_I$: initial configuration

$a_1, a_2, ... \in A$: actions, $\quad c_1, c_2, ... \in C$

$c_0 \overset{a_1 ... a_n}{\longrightarrow} c_n, c_0 \in C_I, c_n \in C_{F_\mathbf{x}}$: valid action sequences V

$$P(c_0 \xrightarrow{a_1} c_1 \mid \mathbf{x}, c_0)$$

$$P(c_1 \xrightarrow{a_2} c_2 \mid \mathbf{x}, c_0 \xrightarrow{a_1} c_1)$$

Transition-Based Parsing

$\mathbf{x} = x_1 ... x_m \in \Sigma^*$: input string,

C : set of configurations, $C_I \subseteq C$: set of intial configurations

$c_0 \in C_I$: initial configuration

$a_1, a_2, ... \in A$: actions, $\quad c_1, c_2, ... \in C$

$c_0 \overset{a_1 ... a_n}{\longrightarrow} c_n, c_0 \in C_I, c_n \in C_{F_\mathbf{x}}$: valid action sequences V

$$P(c_0 \xrightarrow{a_1} c_1 \mid \mathbf{x}, c_0)$$

$$P(c_1 \xrightarrow{a_2} c_2 \mid \mathbf{x}, c_0 \xrightarrow{a_1} c_1)$$

$$P(c_2 \xrightarrow{a_3} c_3 \mid \mathbf{x}, c_0 \xrightarrow{a_1} c_1 \xrightarrow{a_1} c_2)$$

Transition-Based Parsing

$\mathbf{x} = x_1 ... x_m \in \Sigma^*$: input string,

C : set of configurations, $C_I \subseteq C$: set of intial configurations

$c_0 \in C_I$: initial configuration

$a_1, a_2, ... \in A$: actions, $\quad c_1, c_2, ... \in C$

$c_0 \overset{a_1 ... a_n}{\longrightarrow} c_n, c_0 \in C_I, c_n \in C_{F_\mathbf{x}}$: valid action sequences V

$$P(c_0 \overset{a_1}{\longrightarrow} c_1 \mid \mathbf{x}, c_0)$$

$$P(c_1 \overset{a_2}{\longrightarrow} c_2 \mid \mathbf{x}, c_0 \overset{a_1}{\longrightarrow} c_1)$$

$$P(c_2 \overset{a_3}{\longrightarrow} c_3 \mid \mathbf{x}, c_0 \overset{a_1}{\longrightarrow} c_1 \overset{a_1}{\longrightarrow} c_2)$$

$$\vdots$$

$$P(c_n \in C_{F_\mathbf{x}})\ P(c_{n-1} \overset{a_n}{\longrightarrow} c_n \mid \mathbf{x}, c_0 \overset{a_1}{\longrightarrow} c_1 \overset{a_2}{\longrightarrow} ... \overset{a_{n-1}}{\longrightarrow} c_{n-1})$$

Transition-Based Parsing

$\mathbf{x} = x_1 ... x_m \in \Sigma^*$: input string,

C : set of configurations, $C_I \subseteq C$: set of intial configurations

$c_0 \in C_I$: initial configuration

$a_1, a_2, ... \in A$: actions, $\quad c_1, c_2, ... \in C$

$c_0 \xrightarrow{a_1 ... a_n} c_n, c_0 \in C_I, c_n \in C_{F_\mathbf{x}}$: valid action sequences V

$$P(c_0 \xrightarrow{a_1} c_1 \mid \mathbf{x}, c_0)$$

$$P(c_1 \xrightarrow{a_2} c_2 \mid \mathbf{x}, c_0 \xrightarrow{a_1} c_1)$$

$$P(c_2 \xrightarrow{a_3} c_3 \mid \mathbf{x}, c_0 \xrightarrow{a_1} c_1 \xrightarrow{a_1} c_2)$$

$$\vdots$$

$$P(c_n \in C_{F_\mathbf{x}})\ P(c_{n-1} \xrightarrow{a_n} c_n \mid \mathbf{x}, c_0 \xrightarrow{a_1} c_1 \xrightarrow{a_2} ... \xrightarrow{a_{n-1}} c_{n-1})$$

giving $P(o(a_1 ... a_n) \mid \mathbf{x})$ where $o : V \rightarrow GRAPH$

C : set of configurations, $C_I \subseteq C$: set of intial configurations

$c_0 \in C_I$: initial configuration

$a_1, a_2, ... \in A$: actions, $c_1, c_2, ... \in C$

$c_0 \xrightarrow{a_1 ... a_n} c_n, c_0 \in C_I, c_n \in C_I$: valid action sequences V

$$P(c_0 \xrightarrow{a_1} c_1 \mid \quad c_0)$$

$$P(c_1 \xrightarrow{a_2} c_2 \mid \quad c_0 \xrightarrow{a_1} c_1)$$

$$P(c_2 \xrightarrow{a_3} c_3 \mid \quad c_0 \xrightarrow{a_1} c_1 \xrightarrow{a_1} c_2)$$

$$\vdots$$

$$P(c_n \in C_I \quad) \; P(c_{n-1} \xrightarrow{a_n} c_n \mid \quad c_0 \xrightarrow{a_1} c_1 \xrightarrow{a_2} ... \xrightarrow{a_{n-1}} c_{n-1})$$

giving $P(o(a_1 ... a_n) \quad)$ where $o : V \rightarrow GRAPH$

Universal Transition System?

The problem of this paper is to find a **universal** transition system that can produce **arbitrary digraphs**.

> *'I found it quite surprising that there should be a (finite) transition system for arbitrary digraphs.*
> *I would have liked to see a discussion of this seeming contradiction between the present work and the existing literature.*
> *'*
>
> Reviewer

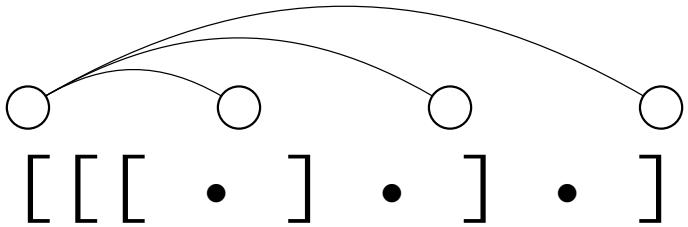
Prior art: no finite axiomatisation for graphs is known.

> *M. Ogawa. Complete axiomatization of an algebraic construction of graphs. In Y. Kameyama and P. J. Stuckey, editors,* Functional and Logic Programming, *pages 163–179, Berlin, Heidelberg. Springer Berlin Heidelberg, 2004*

This paper is rather about **representations** for graphs. Clearly, an adjacency matrix is one possible representation.

Bracketed Representations: Prior Art

Bracketing of Noncrossing Graphs



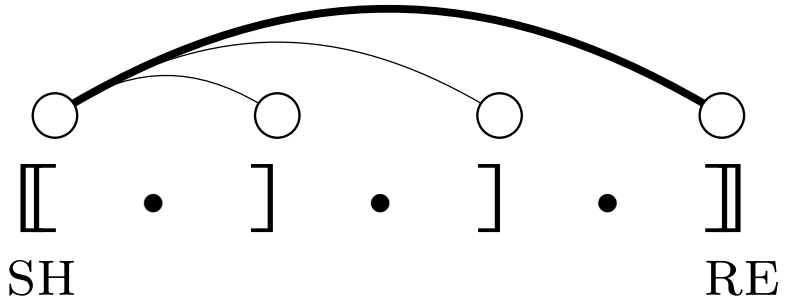
– A. Yli-Jyrä. Approximating dependency grammars through intersection of star-free regular languages. *International Journal of Foundations of Computer Science*, 16(3):565–579, 2005

– A. Yli-Jyrä. On dependency analysis via contractions and weighted FSTs. In D. Santos et al., editors, *Shall We Play the Festschrift Game?, Essays on the Occasion of Lauri Carlson's 60th Birthday*, pages 133–158. Springer, 2012

– A. Yli-Jyrä and C. Gómez-Rodríguez. Generic axiomatization of families of noncrossing graphs in dependency parsing. In *ACL 2017 (Vancouver)*, pages 1745–1755, 2017

Shared Brackets: Prior Art

Bracketing with Shared Endpoints



– A. Yli-Jyrä. Bounded-depth high-coverage search space for noncrossing parses. In *FSMNLP 2017*, pages 30–40, 2017
– A. Yli-Jyrä. How to embed noncrossing trees in Universal Dependencies treebanks in a low-complexity regular language. *Journal of Language Modelling*, 2019. URL: https://doi.org/10.15398/jlm.v7i2.213

Shared Endpoints and Crossing

– E. Pitler et al. Finding optimal 1-endpoint-crossing trees. *TACL*, 1:13–24, 2013
– R. Kurtz and M. Kuhlmann. Exploiting structure in parsing to 1-endpoint-crossing graphs. In *IWPT 2017 (Pisa)*, pages 78–87, 2017
– J. K. Kummerfeld and D. Klein. Parsing with traces: an o(n^4) algorithm and a structural representation. *TACL*, 5:441–454, 2017

The Method

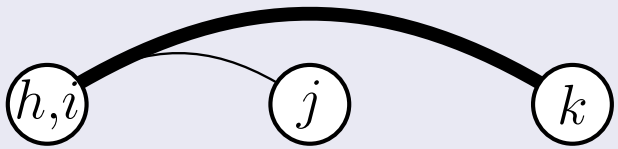
1. **Proper Rope Cover (PRC)**
2. **Rope Decomposition**
3. **Transition System**
4. **Balanced Bracketing**

Preliminaries: Covering Edge

Definition

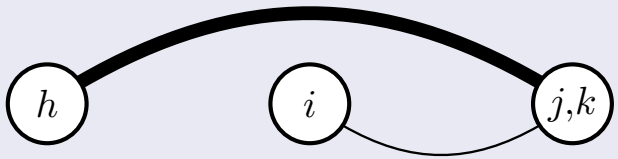
Let $([1, ..., n], E)$ be an ordered graph with $E \subseteq \{(i, j) \mid 1 \leq i < j \leq n\}$. Edge (h, k) is a (properly-longer shared-endpoint) *covering edge* for edge (i, j) if either

- $h = i$ and $i < j < k$



or

- $j = k$ and $h < i < j$.



Denote this situation by $(h, k) : (i, j)$.

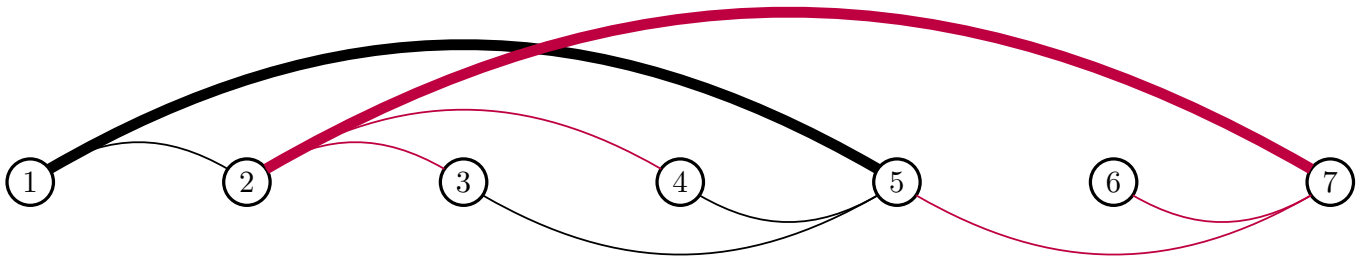
Proper Rope Cover (PRC) – a Novel Concept

Definition

A subset $R \subseteq E$ is a *rope cover* of the graph (V_n, E) if, for every edge $e \in E \backslash R$, there is an edge $c \in R$ such that $c : e$. Moreover, R is a *proper rope cover (PRC)* if is there is no edges $c_1, c_2 \in R$ s.t. $c_1 : c_2$.

Lemma

Every graph has a unique PRC R, and for every $(i, j), (i, k) \in R$, we have $j = k$.



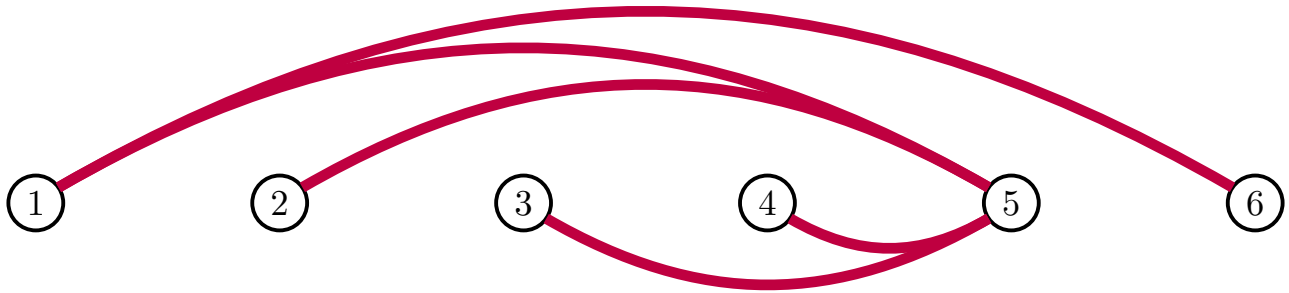
PRC $= \{(1, 5), (2, 7)\}$

Computation of Proper Rope Cover

Lemma

For any graph of n vertices, its PRC and rope-thickness can be computed in $O(n^3)$.

For graph $([6], \{(1,6), (1,5), (2,5), (3,5), (4,5)\})$



we compute PRC as follows:

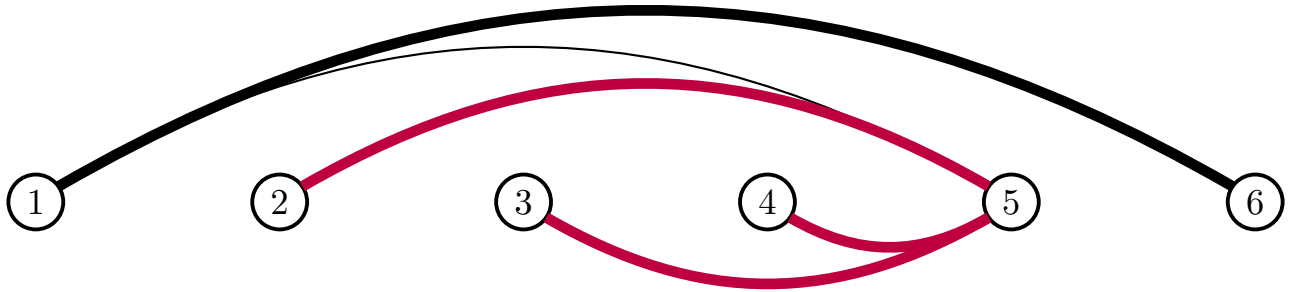
PRC layer	*Remaining edges*
$R_0 = \{\}$	$E_0 = \{(1,6), (1,5), (2,5), (3,5), (4,5)\}$

Computation of Proper Rope Cover

Lemma

For any graph of n vertices, its PRC and rope-thickness can be computed in $O(n^3)$.

For graph $([6], \{(1,6), (1,5), (2,5), (3,5), (4,5)\})$



we compute PRC as follows:

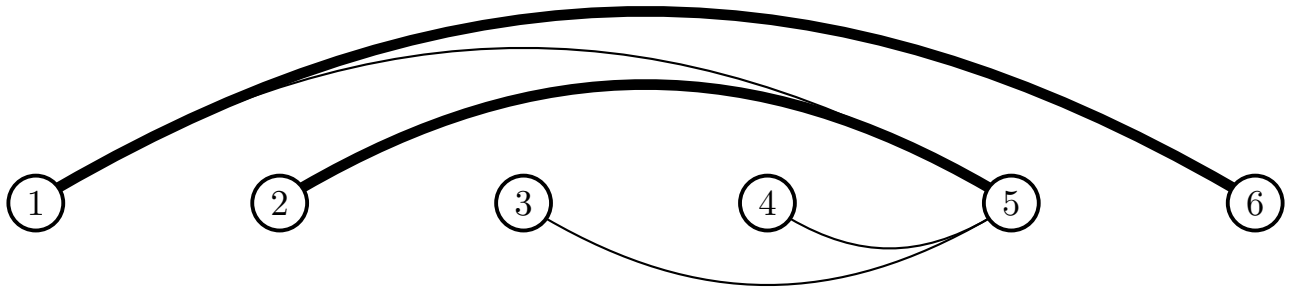
PRC layer	*Remaining edges*
$R_0 = \{\}$	$E_0 = \{(1,6), (1,5), (2,5), (3,5), (4,5)\}$
$R_1 = \{(1,6)\}$	$E_1 = \{(2,5), (3,5), (4,5)\}$

Computation of Proper Rope Cover

Lemma

For any graph of n vertices, its PRC and rope-thickness can be computed in $O(n^3)$.

For graph $([6], \{(1,6), (1,5), (2,5), (3,5), (4,5)\})$



we compute PRC as follows:

PRC layer	*Remaining edges*
$R_0 = \{\}$	$E_0 = \{(1,6), (1,5), (2,5), (3,5), (4,5)\}$
$R_1 = \{(1,6)\}$	$E_1 = \{(2,5), (3,5), (4,5)\}$
$R_2 = \{(2,5)\}$	$E_2 = \{\}$ (algorithm terminates)

The Method

1. **Proper Rope Cover (PRC)**
2. **Rope Decomposition**
3.
4.

Unique Assignment

Theorem

Some edges can have two distinct covering edges in the PRC.

The edge $(2, 3)$ is covered by the edges $(1, 3)$ and $(2, 4)$.

(1)

Unique Assignment

Theorem

Some edges can have two distinct covering edges in the PRC.

The edge $(2, 3)$ is covered by the edges $(1, 3)$ and $(2, 4)$.

(1)

Convention ("EARLIEST")

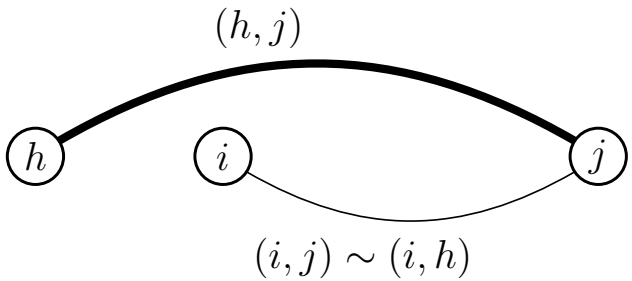
We adopt a convention according to which the ambiguity between two possible covering edges is resolved by assigning the edge to the earliest available covering edge.

Thus, edge $(2, 3)$ is assigned to this EARLIEST covering edge.

Rope Decomposition

Convention ("Indirect Edges")

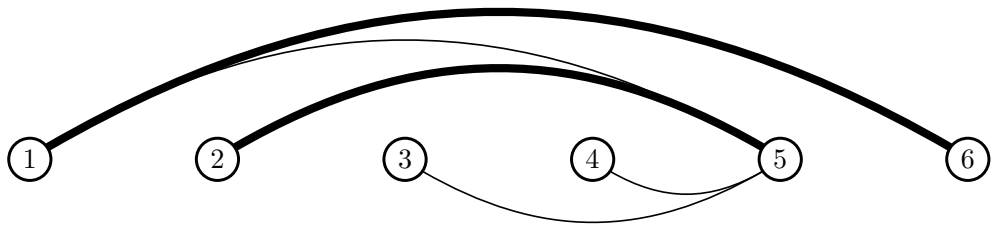
When an edge (i, j) has a covering edge (h, j), $h < i < j$, we refer to the edge (i, j) indirectly, via the pair (i, h) where h is the LEFT INDEX of the covering edge.



Rope Decomposition

Convention ("Indirect Edges")

When an edge (i, j) has a covering edge (h, j), $h < i < j$, we refer to the edge (i, j) indirectly, via the pair (i, h) where h is the LEFT INDEX of the covering edge.



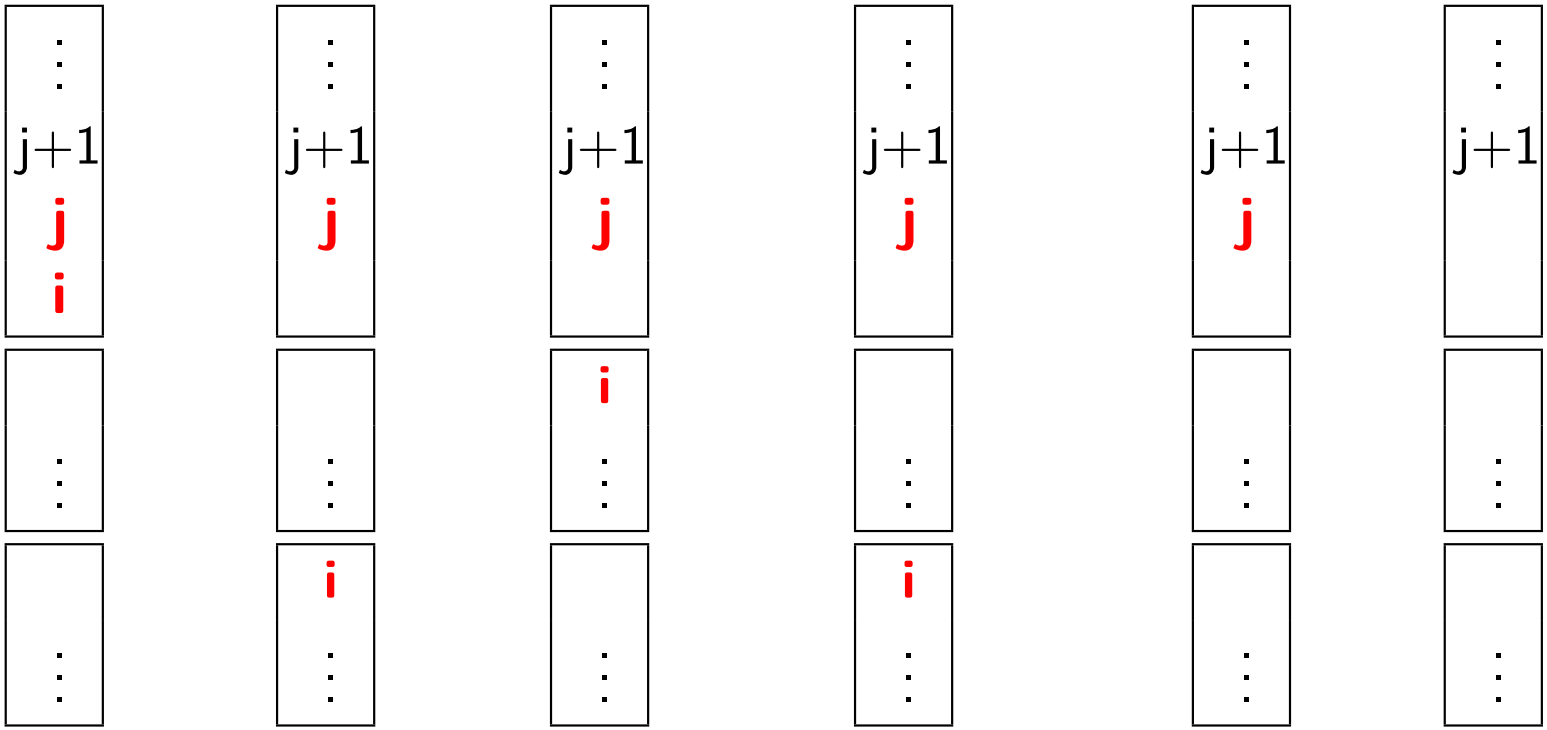
Rope Decomposition of this graph:

- proper rope cover $=\{(1, 6), (2, 5)\}$
- indirect edges $(3, 2)$, $(4, 2)$ referring to edges $(3, 5)$, $(4, 5)$
- other edge $(1, 5)$.

The Method

1. **Proper Rope Cover (PRC)**
2. **Rope Decomposition**
3. **Transition System**
4.

Novel Transition System (Simplified)

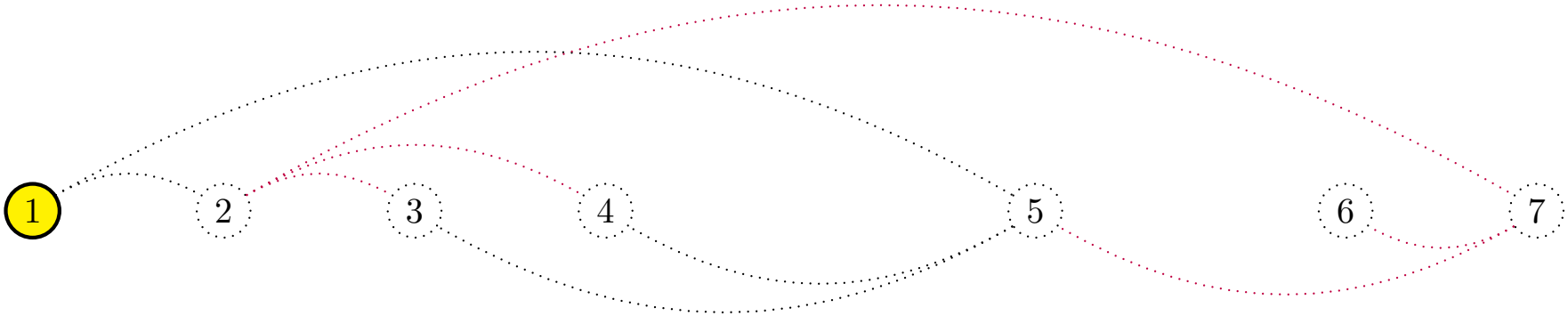


SHift PASS INSert REduce NeXt

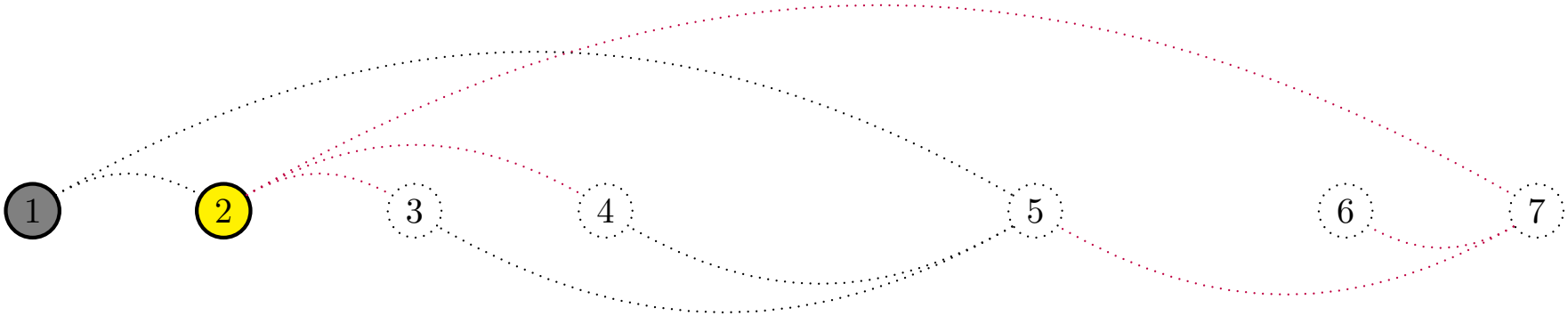
$[\![\bullet$ $]$ $[$ $]\!]$ $\bullet$

In a *digraph*, the edges/some actions have directions: $<$, $>$, $<>$, 0 (dummy action)

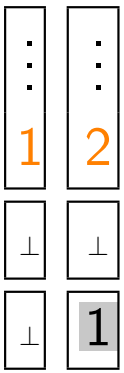
From Transitions to Rope Decomposition



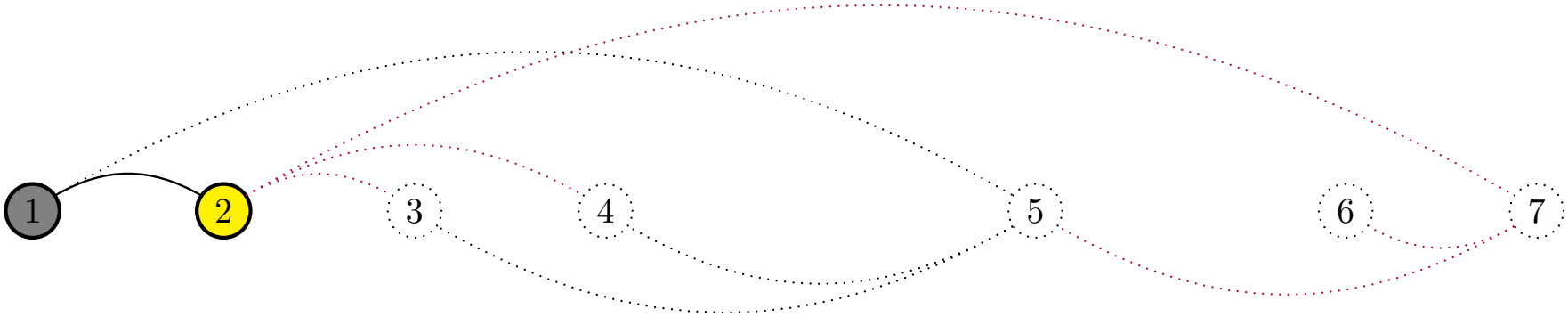
From Transitions to Rope Decomposition



$[\![\bullet$



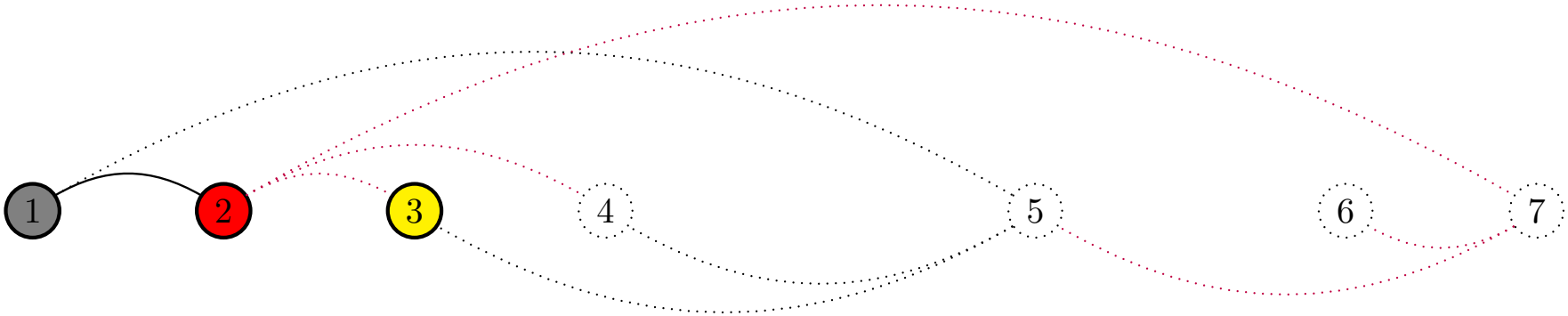
From Transitions to Rope Decomposition

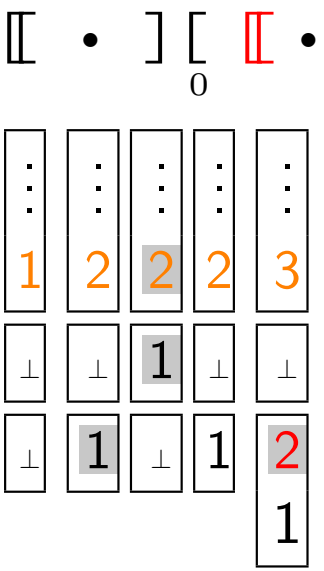


$[\![\; \bullet \;]\!]_0$

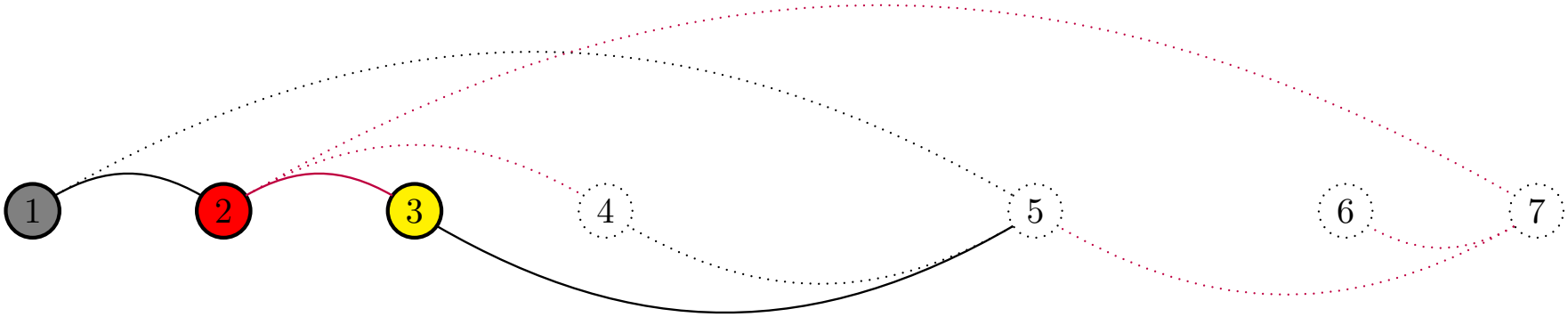
⋮	⋮	⋮	⋮
1	2	2	2
⊥	⊥	1	⊥
⊥	1	⊥	1

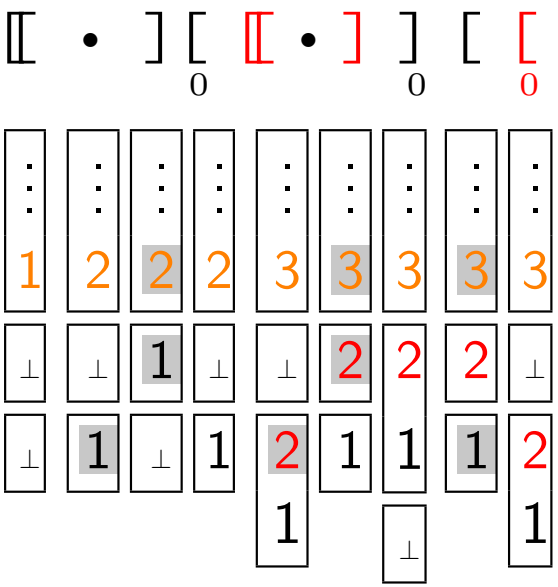
From Transitions to Rope Decomposition



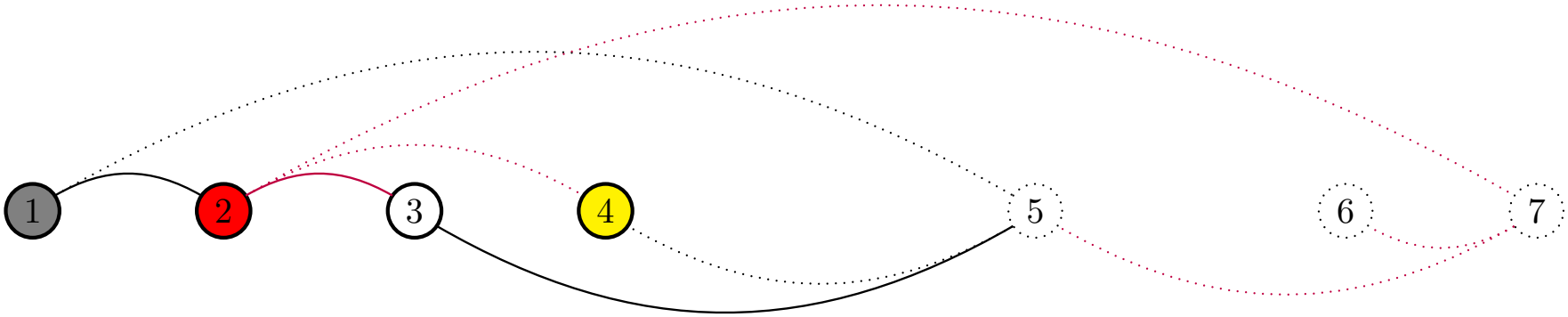


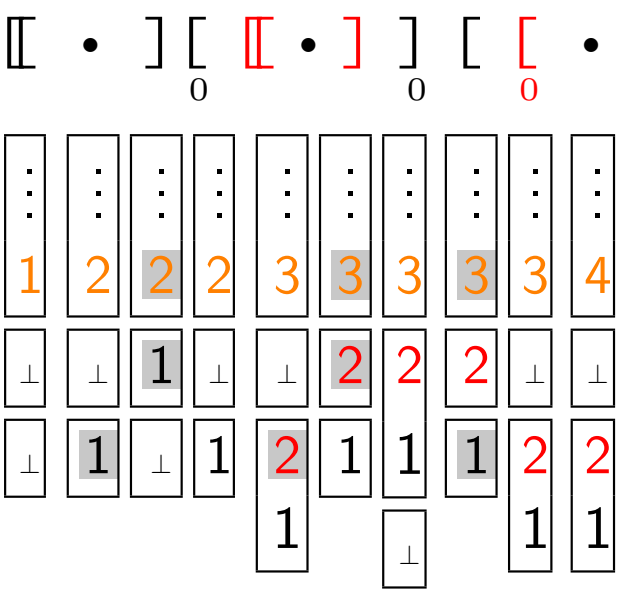
From Transitions to Rope Decomposition



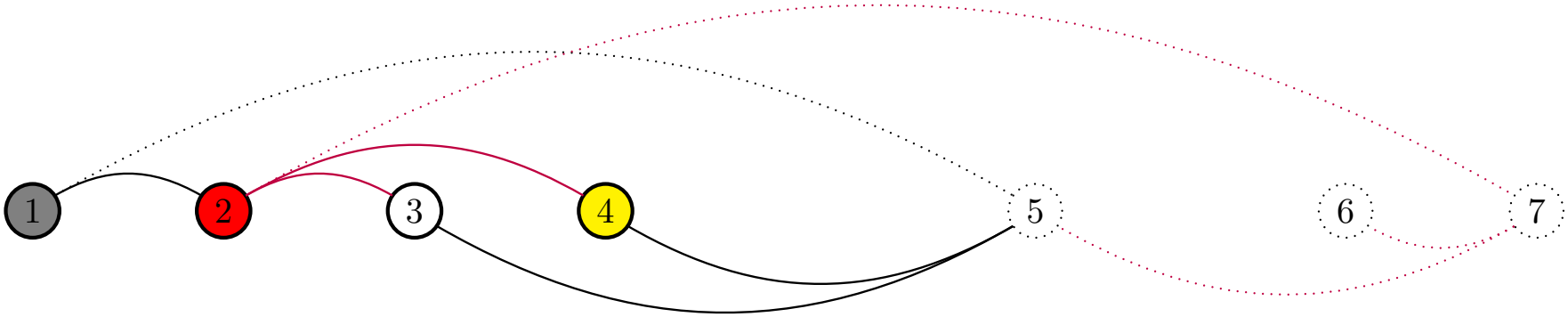


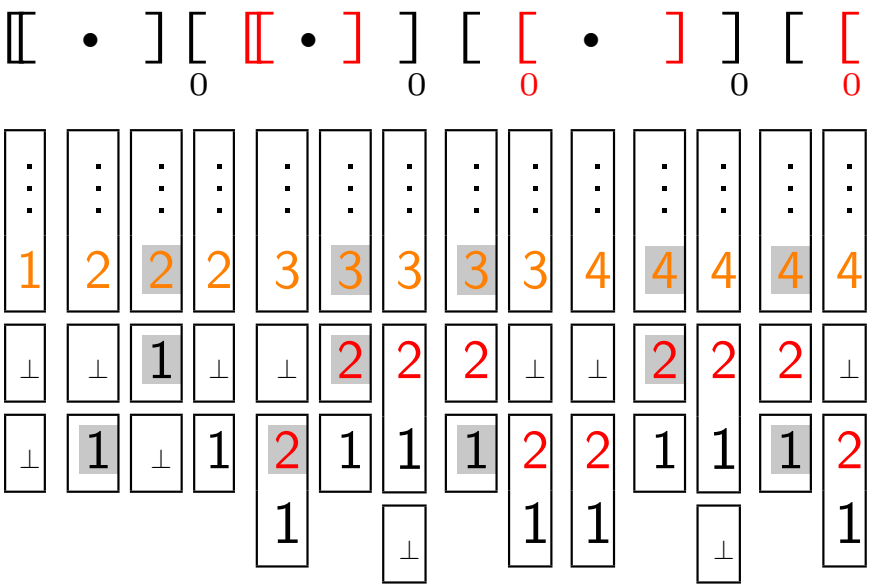
From Transitions to Rope Decomposition



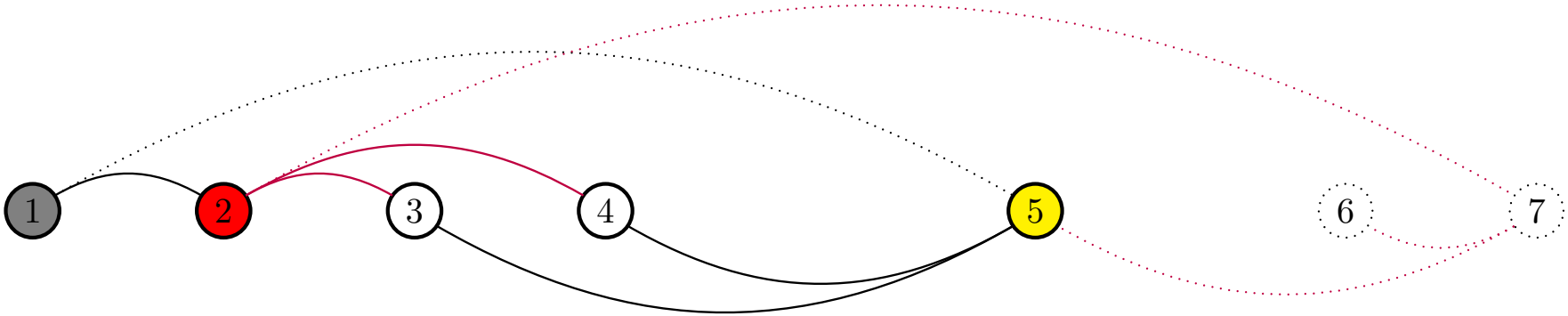


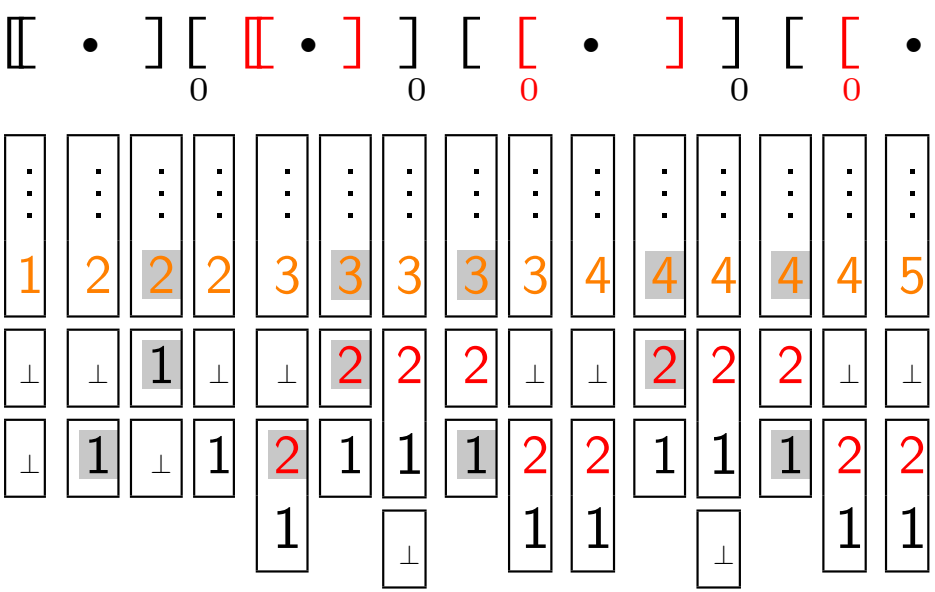
From Transitions to Rope Decomposition



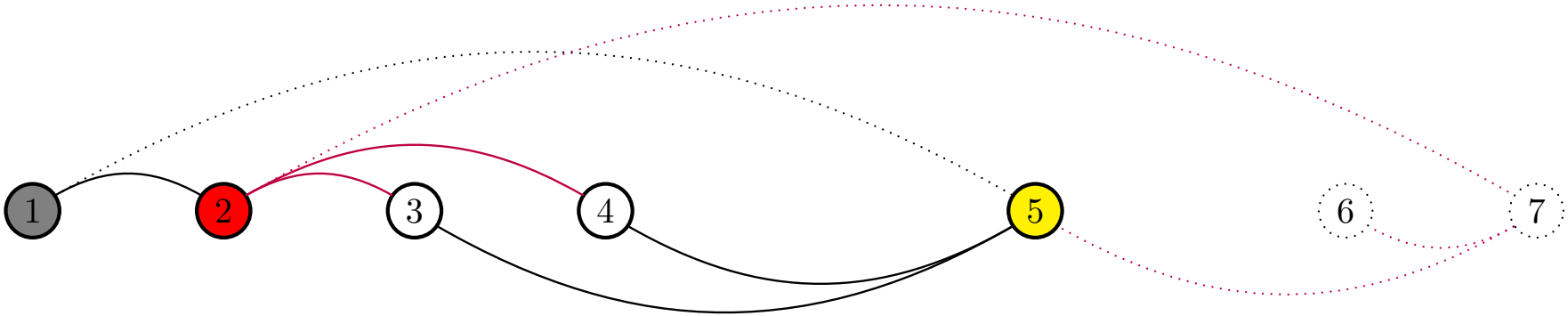


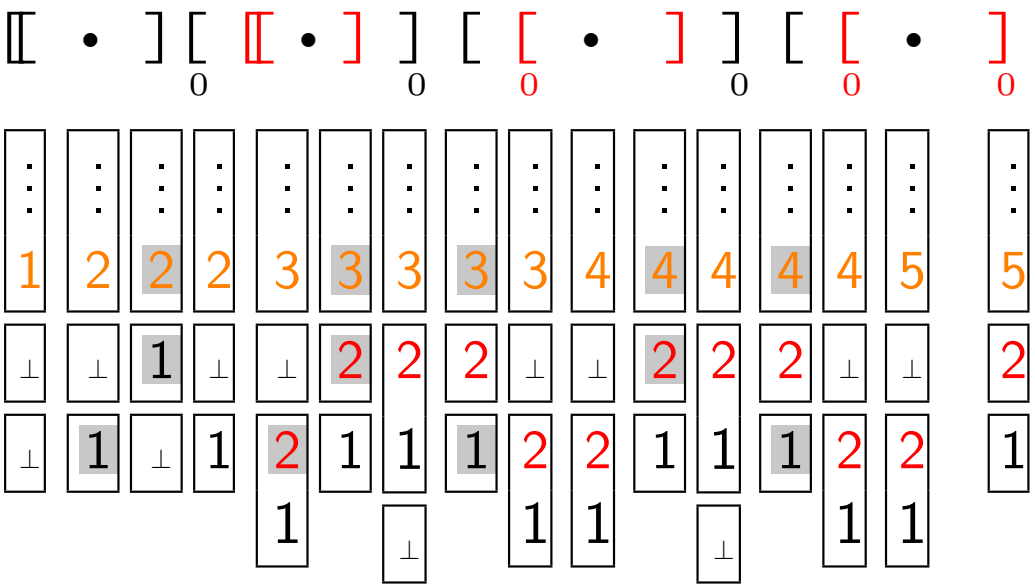
From Transitions to Rope Decomposition



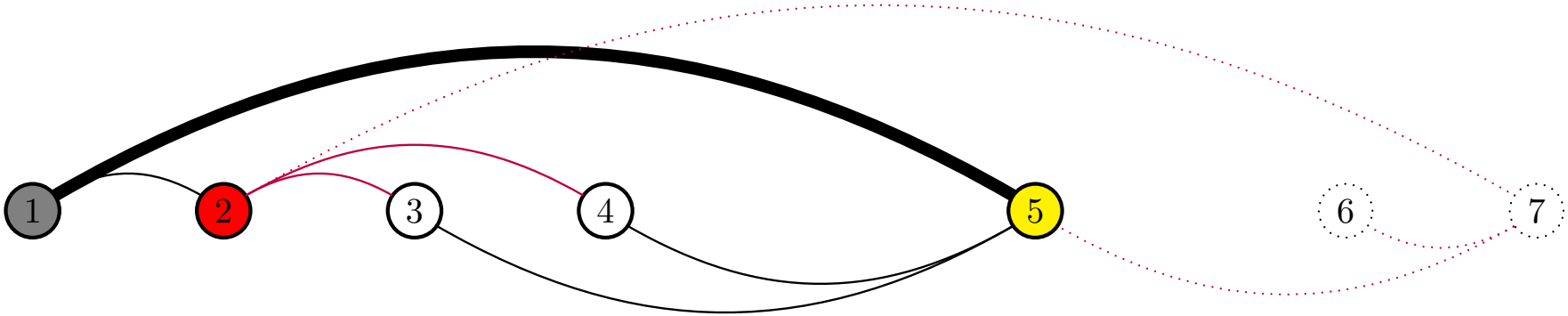


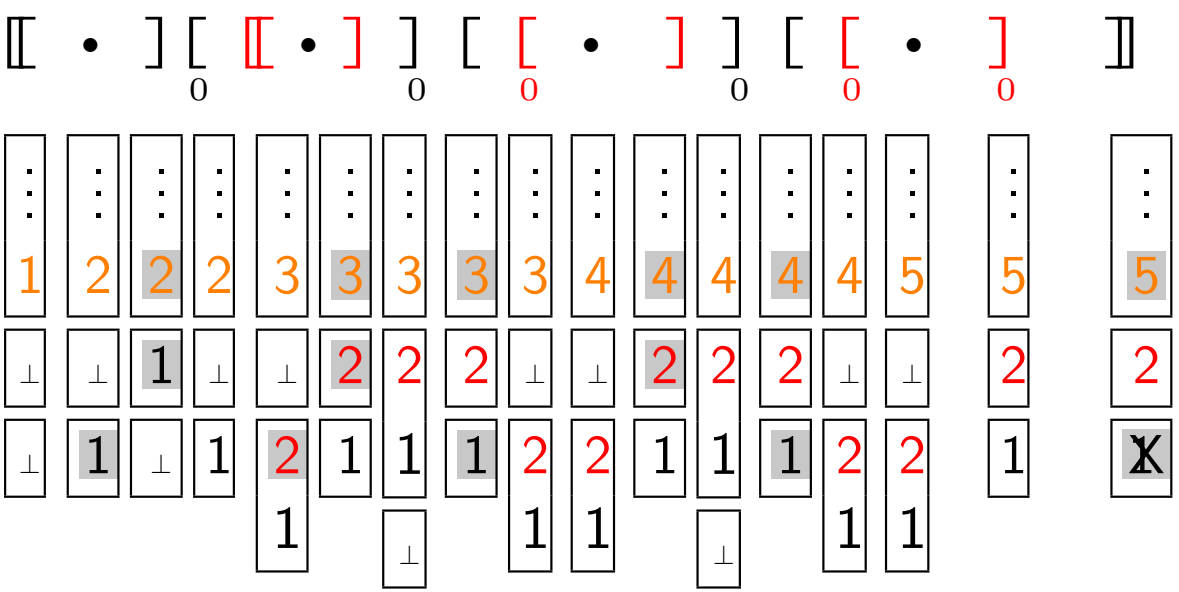
From Transitions to Rope Decomposition



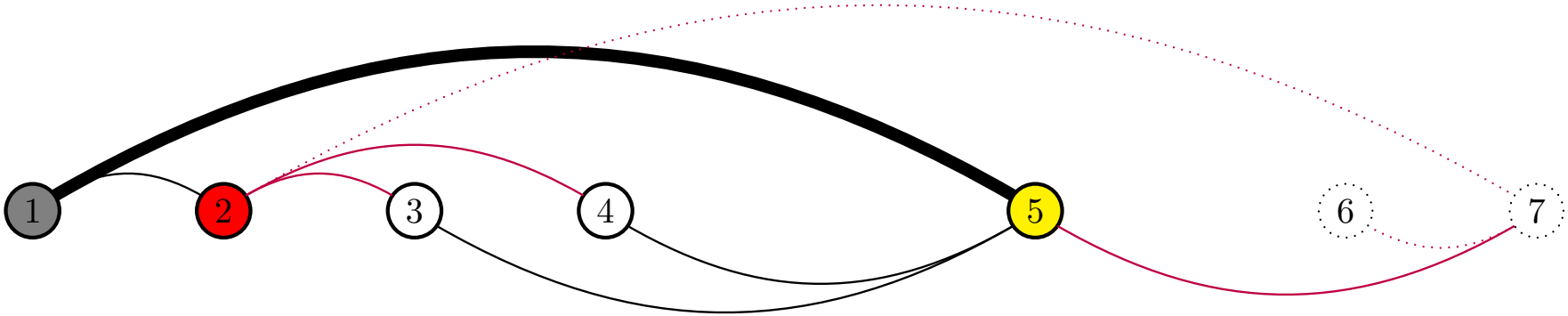


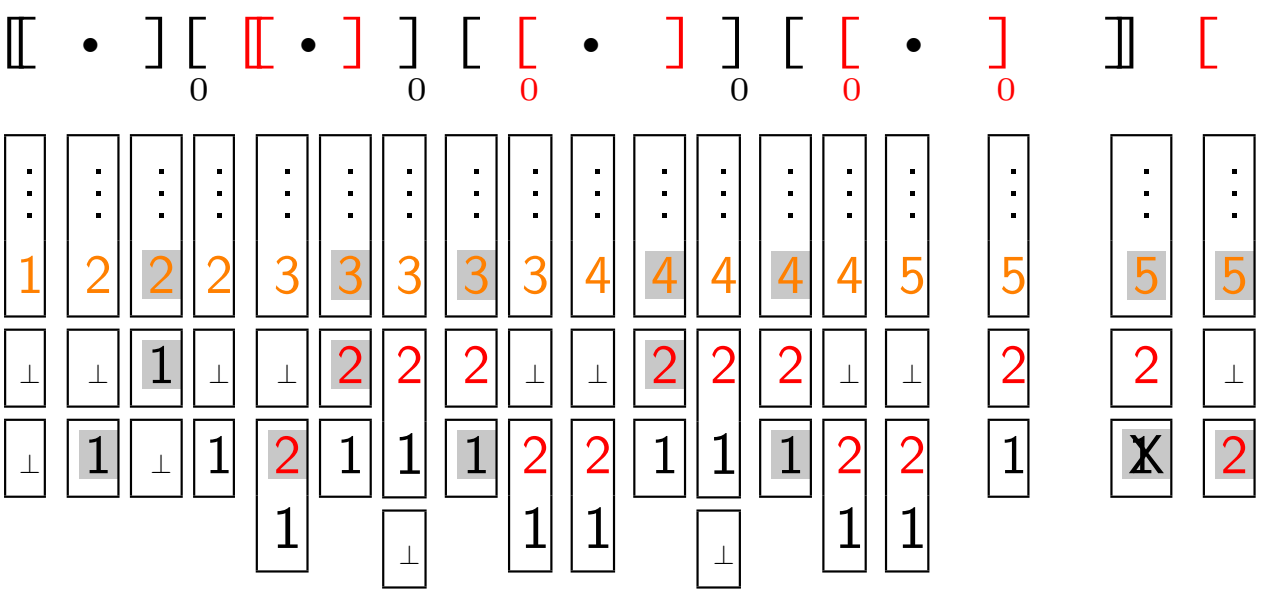
From Transitions to Rope Decomposition



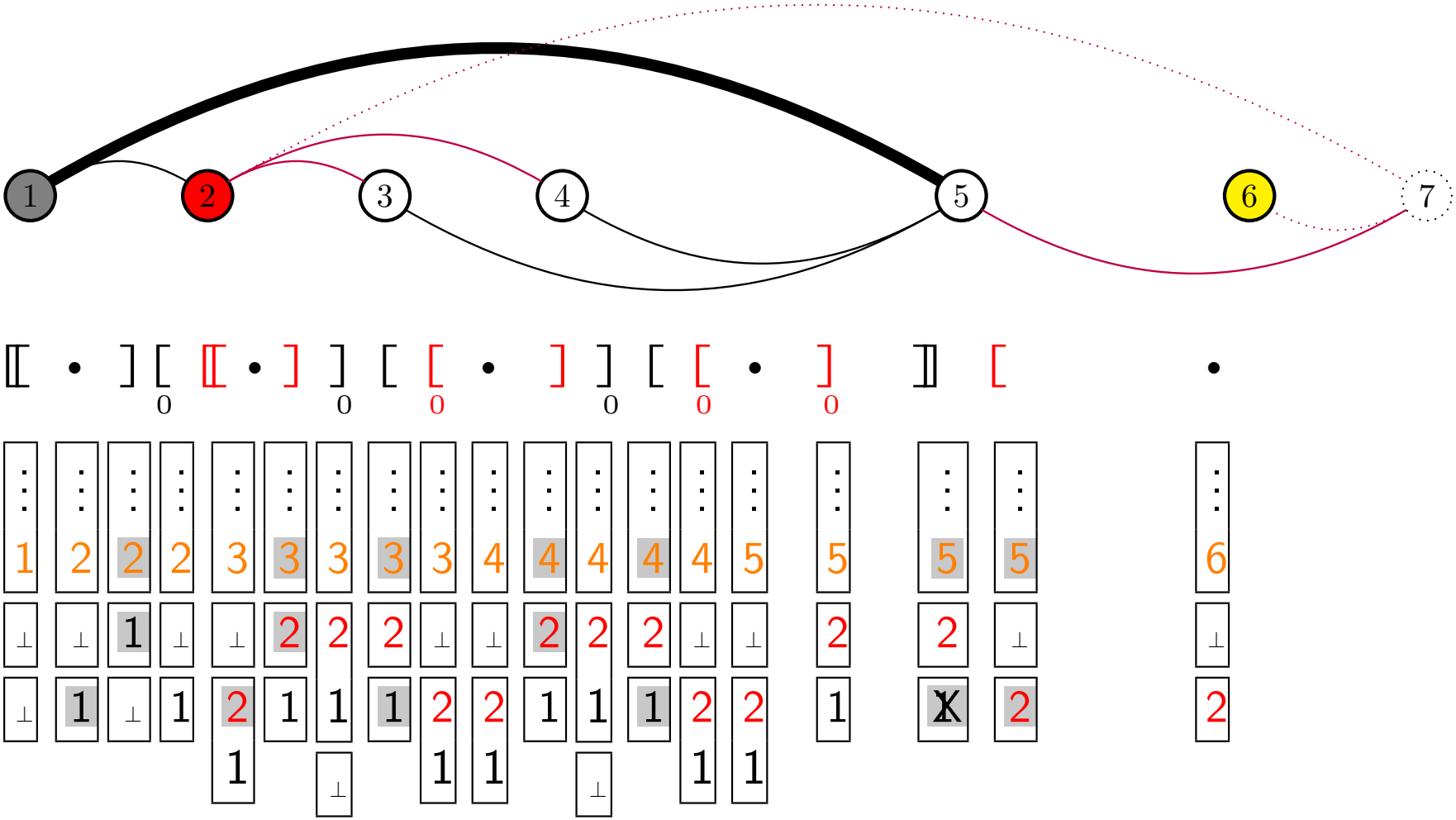


From Transitions to Rope Decomposition

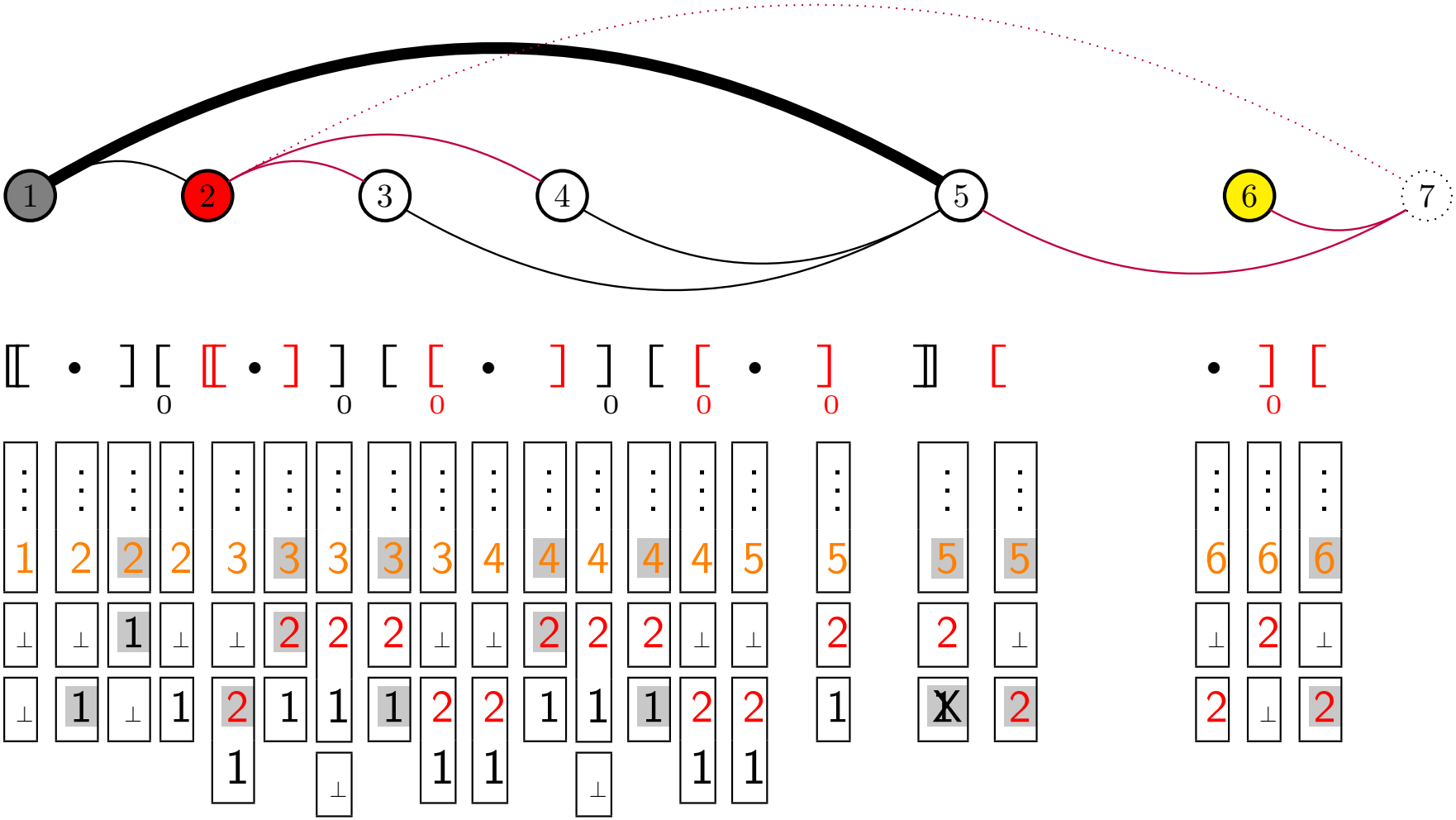




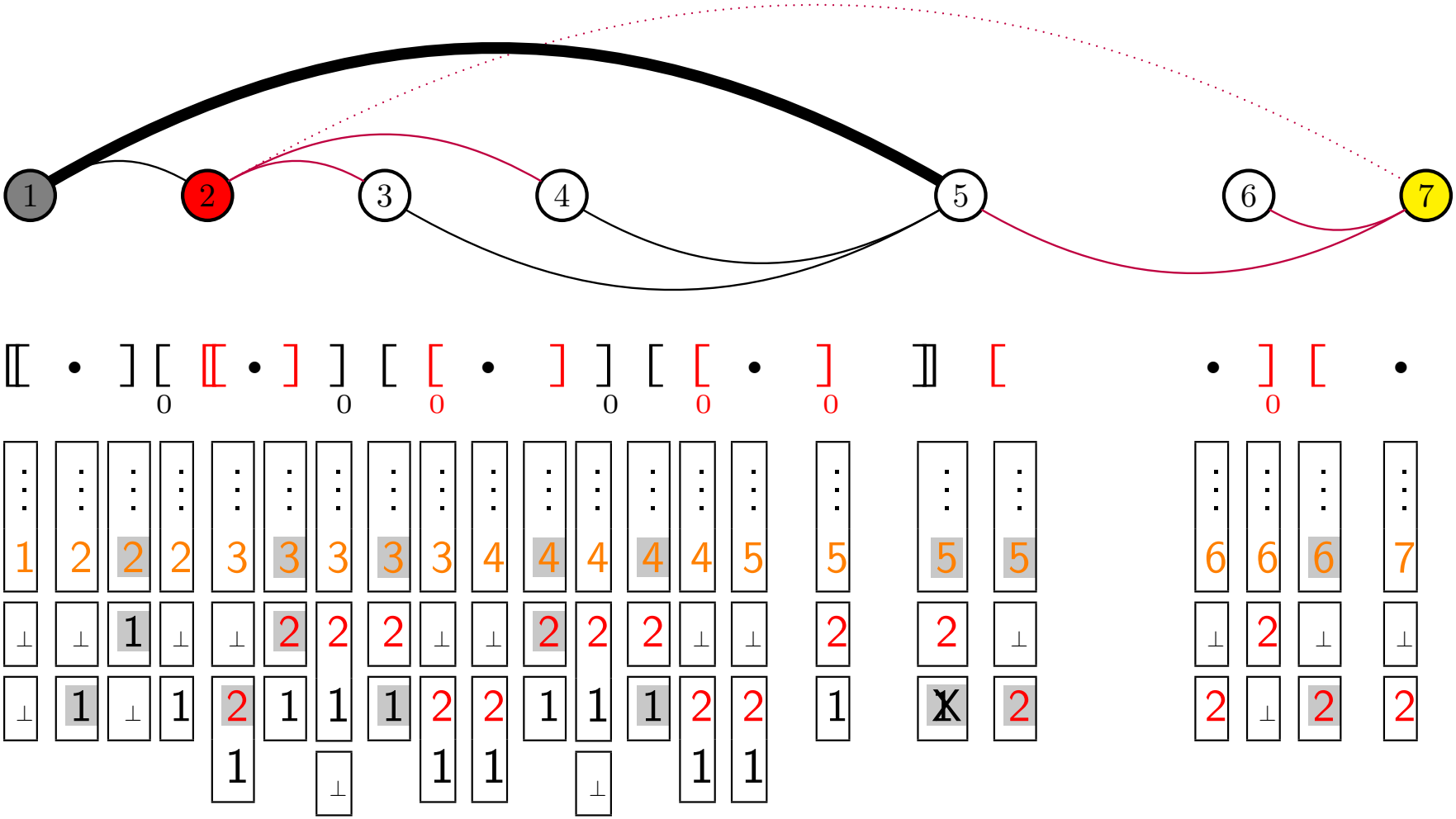
From Transitions to Rope Decomposition



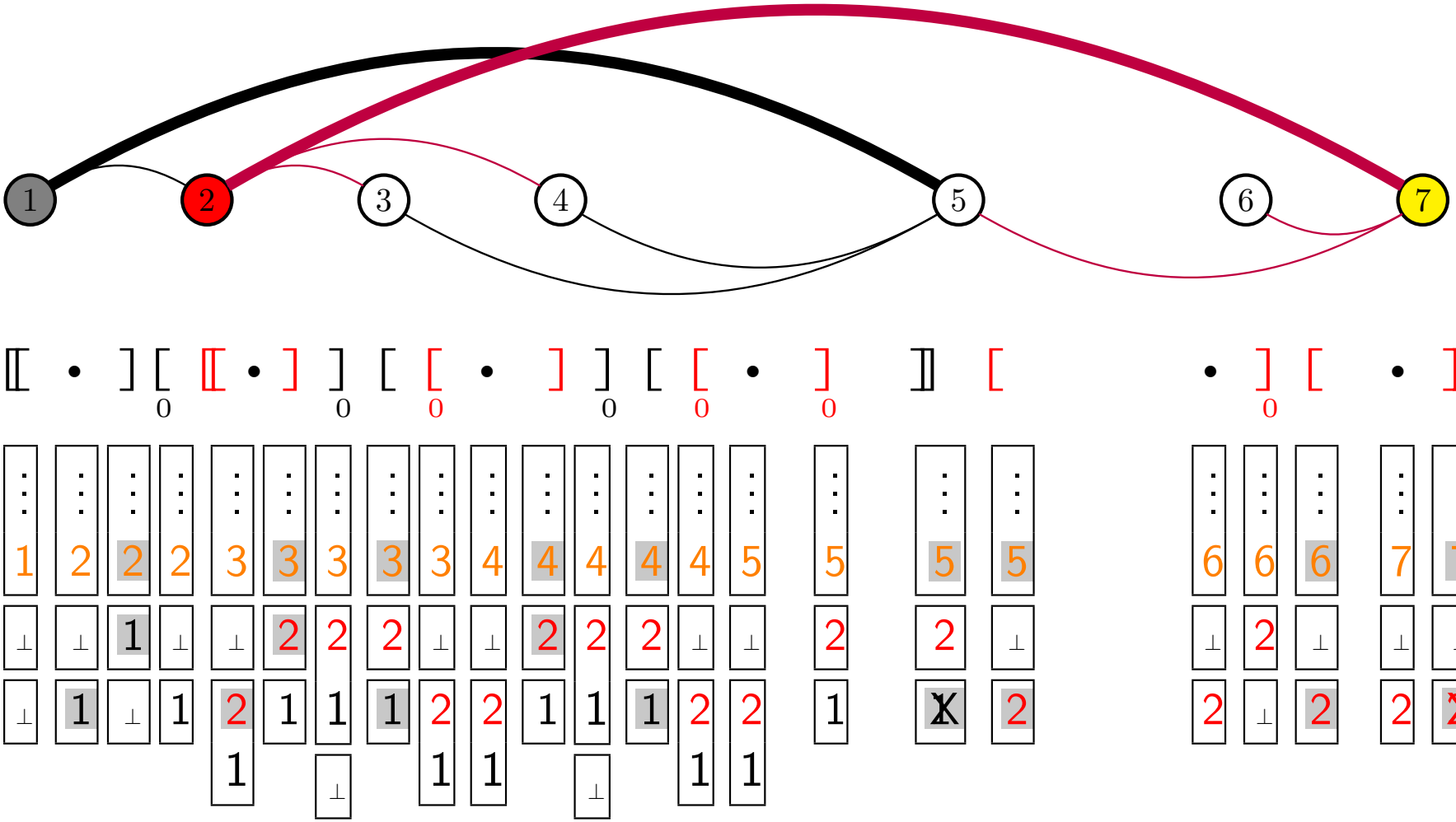
From Transitions to Rope Decomposition



From Transitions to Rope Decomposition



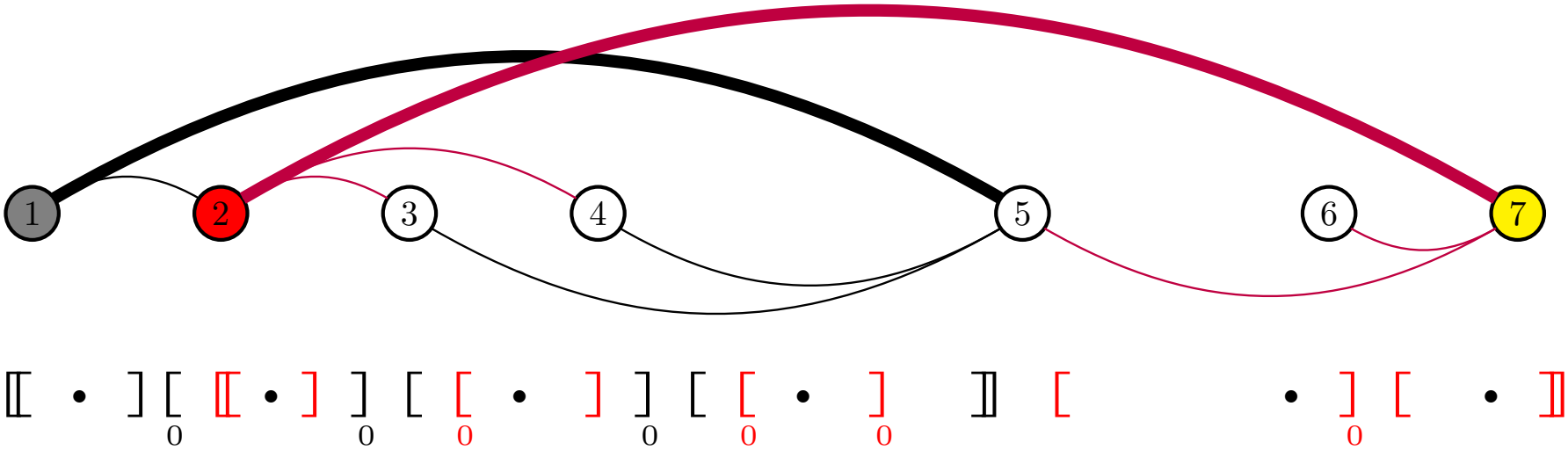
From Transitions to Rope Decomposition



The Method

1. **Proper Rope Cover (PRC)**
2. **Rope Decomposition**
3. **Transition System**
4. **Bracketing**

Bracketing (1/2)



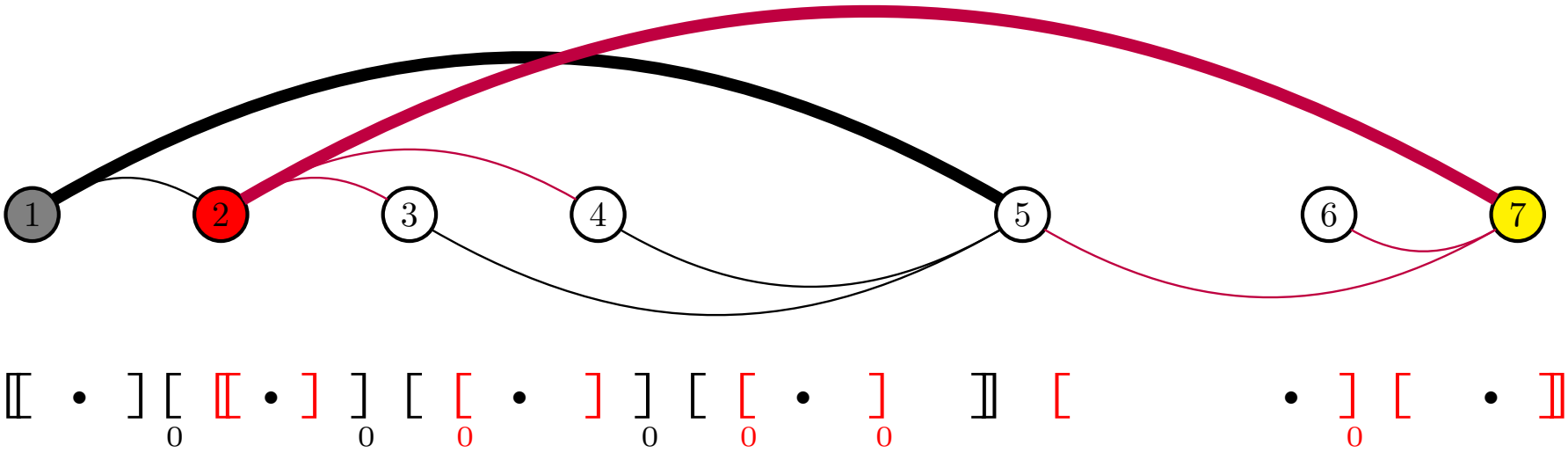
(i) Locally balanced brackets:

$$\mathcal{L} = W \bullet \cdots \bullet W$$

where

$$W = \underbrace{\underset{0}{]}^{*}}_{m} \quad \underbrace{\underset{\mathrm{d}}{]\!]}}_{\text{optional}} \quad \underbrace{\underset{\mathrm{d}}{]}^{*}}_{n} \quad \underbrace{\underset{\mathrm{d}}{[}^{*}}_{m+n} \quad \underbrace{[\![}_{\text{optional}}$$

Bracketing (2/2)



(ii) Globally balanced brackets

$$S \rightarrow \epsilon \mid \bullet \mid S\,S \mid \llbracket S \rrbracket_{d_1} \mid [\,S\,]_{d_1}^{d_1} \mid [\,S \rrbracket_{d_2}^{e} \mid \llbracket S \rrbracket_{e}$$

where

$$d_1, d_2 \in \{<, >, <>, 0\}, e \in \{<, >, <>\}$$

The Results

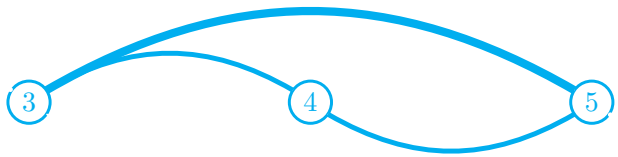
1. Relevance to natural language complexity
2. Bijection $\mathrm{DIGRAPH} \leftrightarrow L_{\mathrm{DIGRAPH}}$
3. Subsets in formal language theory
4. Succinct graph representation

Rope-Thickness

Definition

For graph (V_n, E) with a PRC R, the *rope-thickness of a vertex* $i \in V_{n-1}$ is the number of edges $(h, j) \in R$ satisfying $h \leq i < j$. The *rope-thickness of the graph* is the maximum over the rope-thicknesses of all vertices $i \in V_{n-1}$ in the graph.

Example. Complete graph of 3 vertices and 3 edges:



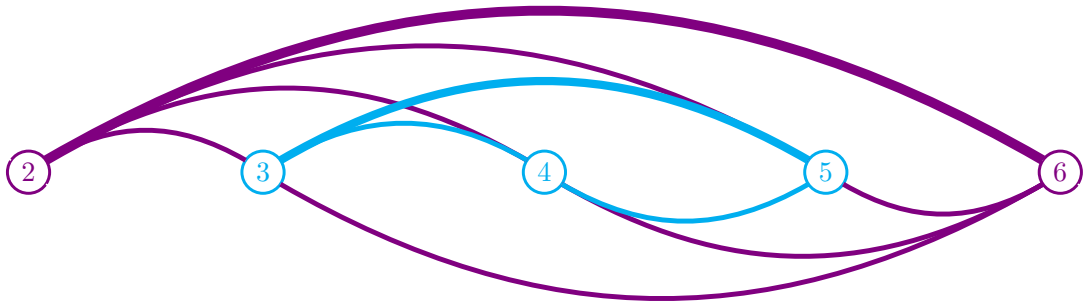
This graph has rope thickness 1

Rope-Thickness

Definition

For graph (V_n, E) with a PRC R, the *rope-thickness of a vertex* $i \in V_{n-1}$ is the number of edges $(h, j) \in R$ satisfying $h \leq i < j$. The *rope-thickness of the graph* is the maximum over the rope-thicknesses of all vertices $i \in V_{n-1}$ in the graph.

Example. Complete graph of 3+2 vertices and 3+7 edges:



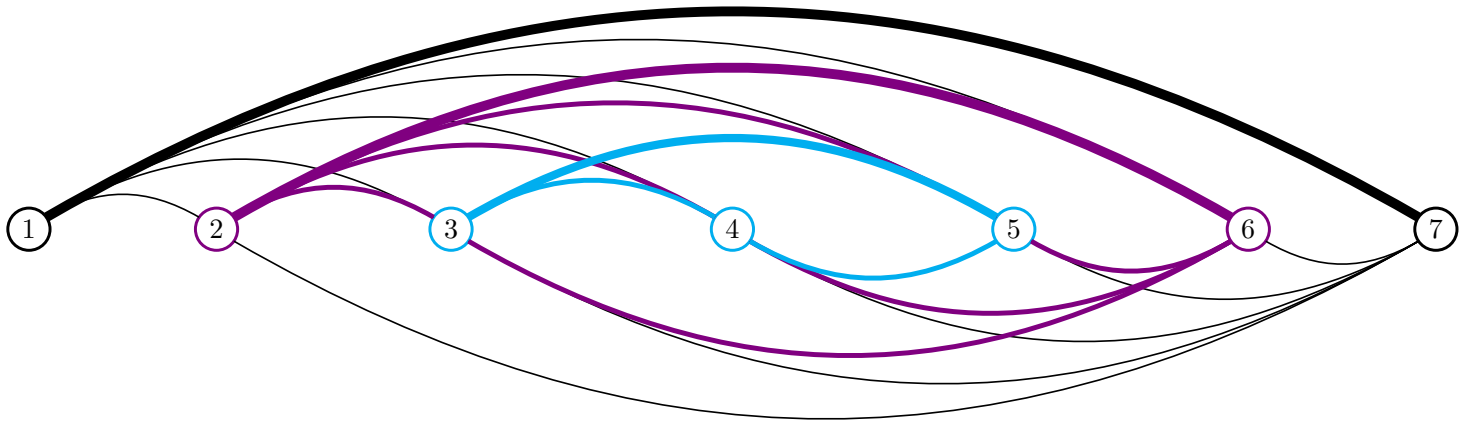
This graph has rope thickness 1+1

Rope-Thickness

Definition

For graph (V_n, E) with a PRC R, the *rope-thickness of a vertex* $i \in V_{n-1}$ is the number of edges $(h, j) \in R$ satisfying $h \leq i < j$. The *rope-thickness of the graph* is the maximum over the rope-thicknesses of all vertices $i \in V_{n-1}$ in the graph.

Example. Complete graph of 3+2+2 vertices and 3+7+11 edges:



This graph has rope thickness 1+1+1

Rope-Thickness in Universal Dependencies v2.4

language	trees	rope-th. 1	2	3	4	5	6	7	8
Arabic	28,402	4.9%	20.5%	63.5%	94.0%	99.64%	99.986%	100.000%	100.000%
Czech	127,507	13.3%	48.3%	89.3%	98.8%	99.91%	99.992%	100.000%	100.000%
Dutch	20,735	21.3%	63.4%	92.9%	98.9%	99.92%	99.986%	100.000%	100.000%
English	34,601	15.6%	54.9%	92.3%	99.3%	99.98%	99.997%	100.000%	100.000%
Finnish	34,641	37.0%	78.7%	96.8%	99.6%	99.97%	99.994%	100.000%	100.000%
German	191,757	10.1%	49.2%	90.5%	99.2%	99.96%	99.997%	100.000%	100.000%
Hindi	19,545	2.9%	44.4%	85.3%	98.3%	99.87%	99.980%	100.000%	100.000%
Italian	34,057	5.6%	49.9%	90.3%	98.8%	99.89%	99.997%	100.000%	100.000%
Korean	34,702	15.5%	65.6%	94.7%	99.5%	99.95%	99.994%	100.000%	100.000%
Latvian	12,558	12.7%	50.0%	89.8%	98.8%	99.90%	99.992%	99.992%	100.000%
Russian	87,377	15.0%	55.5%	90.7%	98.9%	99.91%	99.994%	99.999%	100.000%
Chinese	18,628	46.1%	73.7%	91.9%	98.6%	99.89%	99.989%	100.000%	100.000%
all 83	1,232,262	15.1%	54.8%	90.5%	99.0%	99.93%	99.995%	100.000%	100.000%

Processing the UD v2.4 trees requires only finite-state memory!
Preliminary results on graph banks are similar.

Certainly prompts further research on language complexity

Encoded Graph Languages

Given

- $\mathcal{L}$ – $\bullet$-marked Kleene star of W, the vertex bracketing lexicon
- $\mathcal{D}$ – the Dyck language of balanced brackets in encoded graphs

We obtain **language representations à la Chomsky and Schützenberger**

1. All ordered digraphs

$$L_{\mathrm{DIGRAPH}} = h(h^{-1}(\mathcal{L}) \cap \mathcal{D}) \in \mathcal{INDEXED}$$

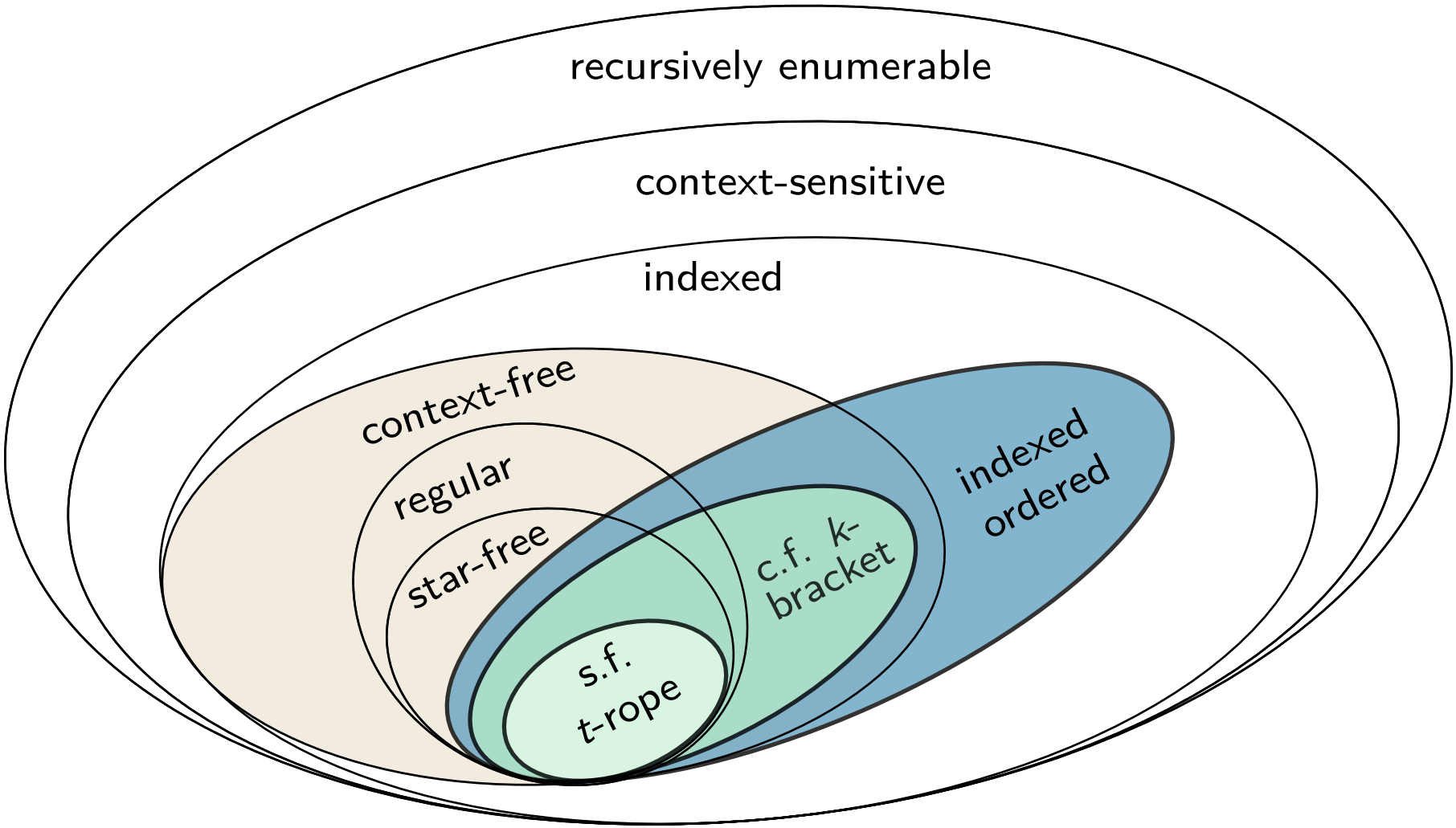
2. k-bounded brackets in vertex lexicon

$$L_{\mathrm{DIGRAPH},k} = h(h^{-1}(\mathcal{L}_{<k}) \cap \mathcal{D}) \in \mathcal{CF}$$

3. t-bounded rope thickness

$$L_{\mathrm{DIGRAPH},2t,t} = h(h^{-1}(\mathcal{L}_{<2t}) \cap \mathcal{D}_t) \in \mathcal{REGULAR}$$

String Language Subsets of All Encoded Digraphs



Succinct Graph Representation

Definition (Turán 1984)

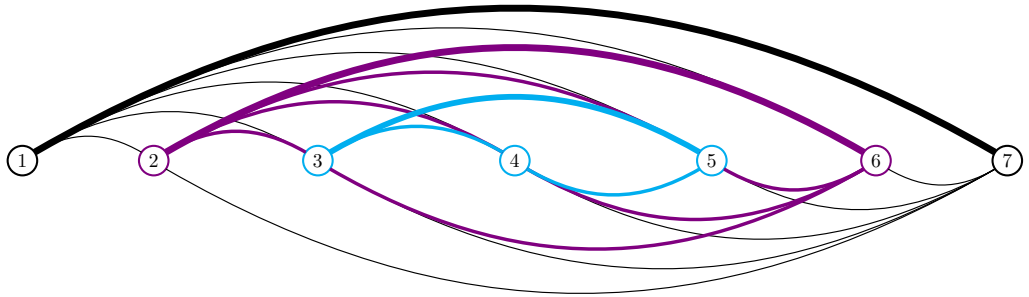
Given a class C of graphs how many bits and how much time are needed to encode a graph $G \in C$ into a binary string s, and to decode s to reconstruct G?

class of graphs	bits	author
planar graph	$3.58m$	Keeler&Westbrook 1995
planar graph	$2m+8n+o(n)$	Munro&Raman 1997
planar graph	$2m+(5+1/k)n+o(n)$	Chuang et al. 1998
planar graph	$2m+2n+o(n)$	Chiang,Lin,Lu 2005
arb. graphs	$1.5n^2+3n$	**this work**
arb. digraphs	$2n^2+4n$	**this work**
arb. digraphs	$lg \binom{n^2}{m}$	an inf. theoretic lower bound (Farzan&Munro 2013)
arb. graphs	$n(n-1)/2$	numbering, log of A006125
arb. digraphs	$n(n-1)$	numbering, log of A000088

Now $1.5n^2 + 3n$b/compl.graph, $2n^2 + 4n$b/compl.digraphs

Proposition

The current representation needs $O(n^3)$ encoding time, $O(n + |E|)$ decoding time, and at most $1.5n^2 + 3n$ bits (with block code).



$[[\bullet][[\bullet]][[[\bullet]]][[[\bullet]]][[\bullet]][\bullet]]$(*eos*)

symb	count	bits		
$]/]_0/[/[_0$	18	54	$6\lfloor n/2 \rfloor \lceil n/2 - 1 \rceil$	
$]]$	3	9	$3\lfloor n/2 \rfloor$	
$[[\bullet/\bullet$	6	18	$3n - 3$	
(eos)	1	3	3	
total	28	84	$3\lfloor n/2 \rfloor (2\lceil n/2 \rceil - 1) + 3n$	$\approx 1.5n^2 + 3n$

Useful for Parsing?

'Does the (context-freeness of the) encoding provide access to methods and results from classical formal language theory that can be used to solve problems in, say, non-projective dependency parsing more efficiently or elegantly than by working directly on the graphs?' a reviewer

Useful for Parsing?

'... these rope decompositions look related to the endpoint-crossing definitions...' a colleague in NLP, p.c.

- Turns string FSTs into string-to-graph transducers
- $O(n^3)$ parsing for noncrossing graphs:
  - Chomsky-Schützenberger parsing
  - Parsing by contractions
- Neural parsing
  - Supertagging, parsing as labeling
  - Encoder-decoder parsing

Conclusion

- **Problem:** transition system for all graphs?
- **New method:** *proper rope cover* – unique and minimal
- **Results:** universal transition system
  - empirical relevance: 6-rope covers 99.995% of UD v2.4
  - succinct representation $\mathrm{DIGRAPH} \leftrightarrow L_{\mathrm{DIGRAPH}}$
  - **conllenc.py** at `https://github.com/amikael/coding`
- Related to existing parsing frameworks
- Further work
  - decidable MSO logic fragments over graphs
  - semantic graphs
  - maximum subgraph problems
  - enumerative graph theory