Balanced-by-construction regular and ω-regular languages

Luc Edixhoven Sung-Shik Jongmans

Open University of the Netherlands

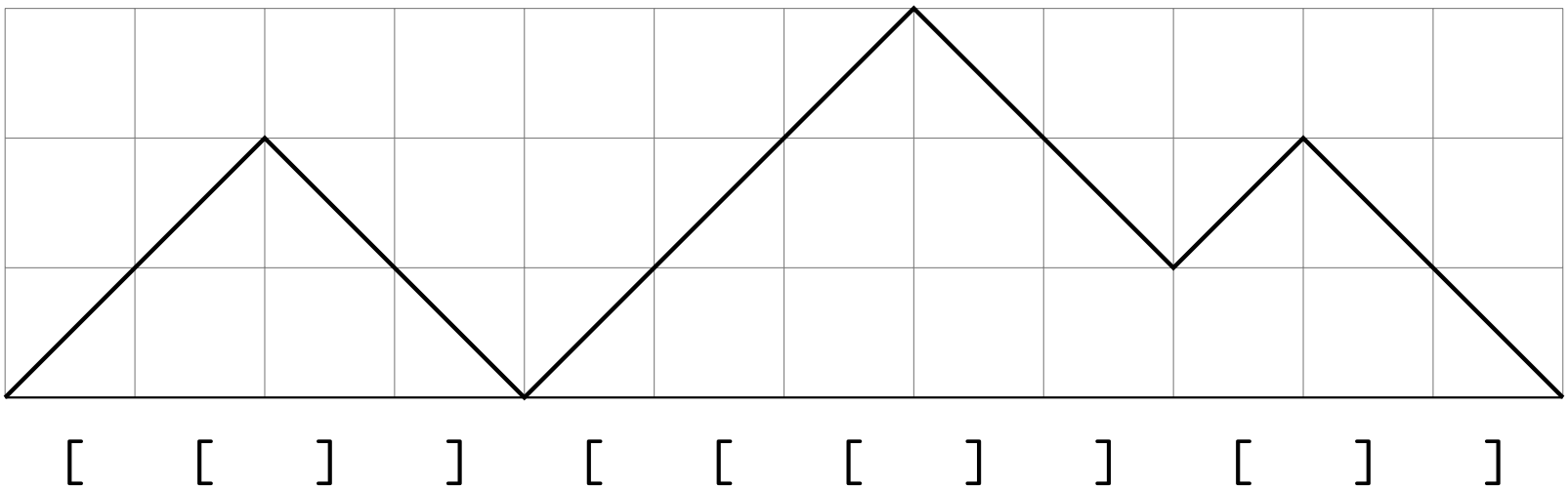
Centrum Wiskunde & Informatica (CWI)

DLT 2021

Outline

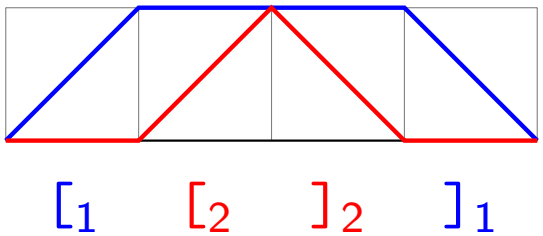
- Balancedness
- Recognising balanced regular languages
- Expressing balanced regular languages
- ω-regular languages

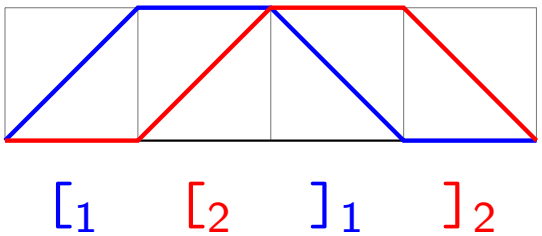
Balancedness



Dyck language: balanced parentheses

Balancedness



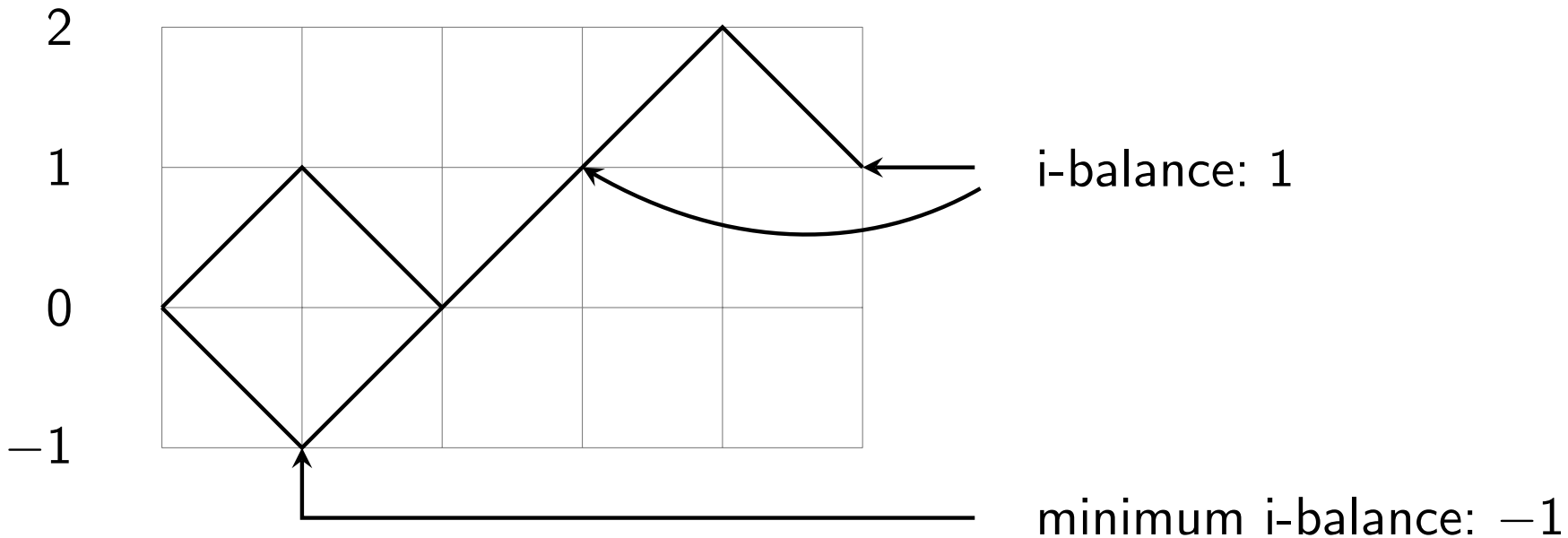


Paren_n ✓ ✗

Balanced ✓ ✓

Recognising balanced regular languages

$$(\,[_i]_i +]_i[_i)[_i([_i]_i + \lambda)$$



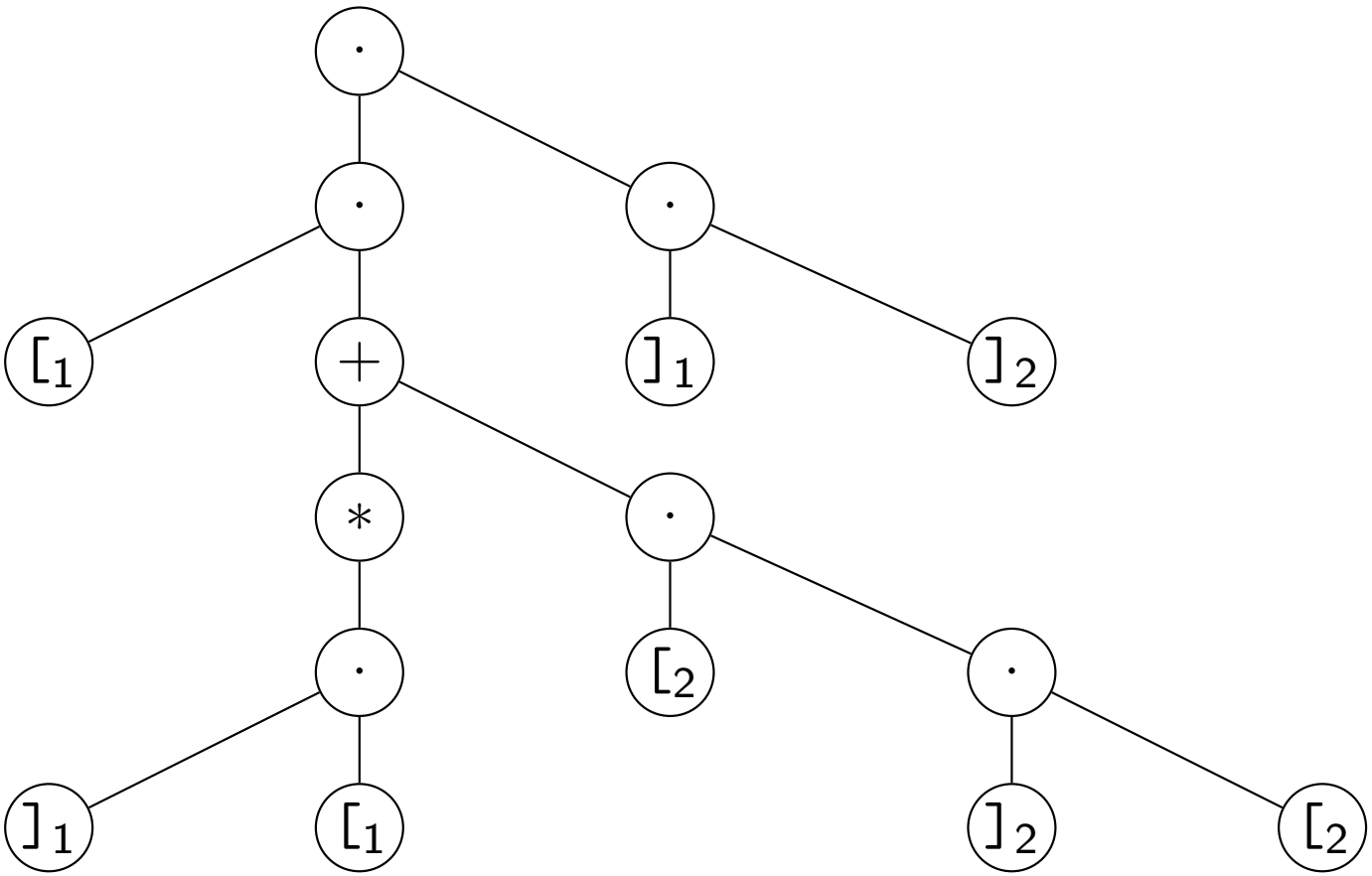
For the i-balance to be unique:

- In $e_1 + e_2$, both branches should have the same i-balance
- In e^*, e should have an i-balance of 0

Recognising balanced regular languages

$e = [_1((]_1[_1)^* + [_2]_2[_2)]_1]_2$

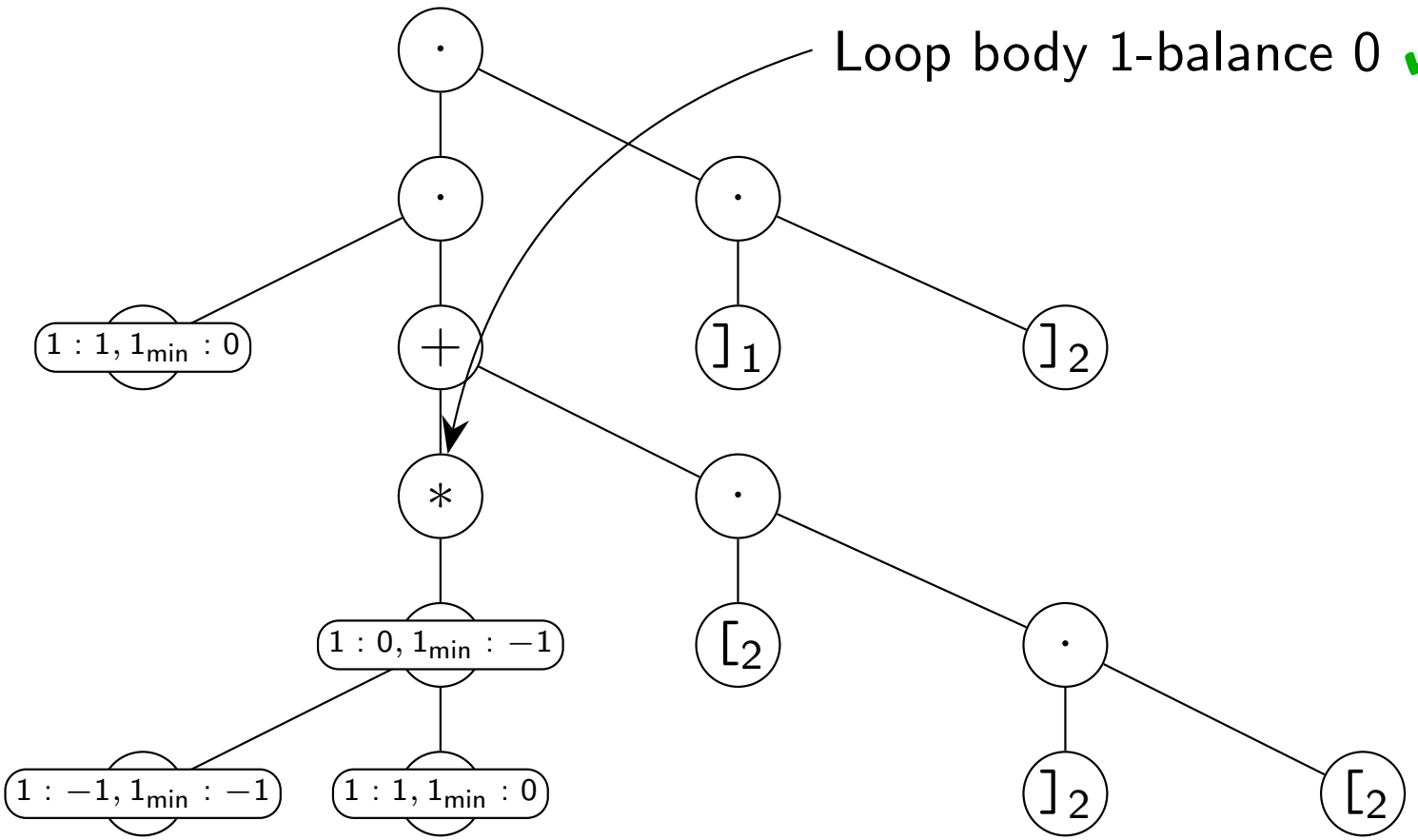
Checking the 1-balance



Recognising balanced regular languages

$e = [_1([]_1[_1)^* + [_2]_2[_2)]_1]_2$

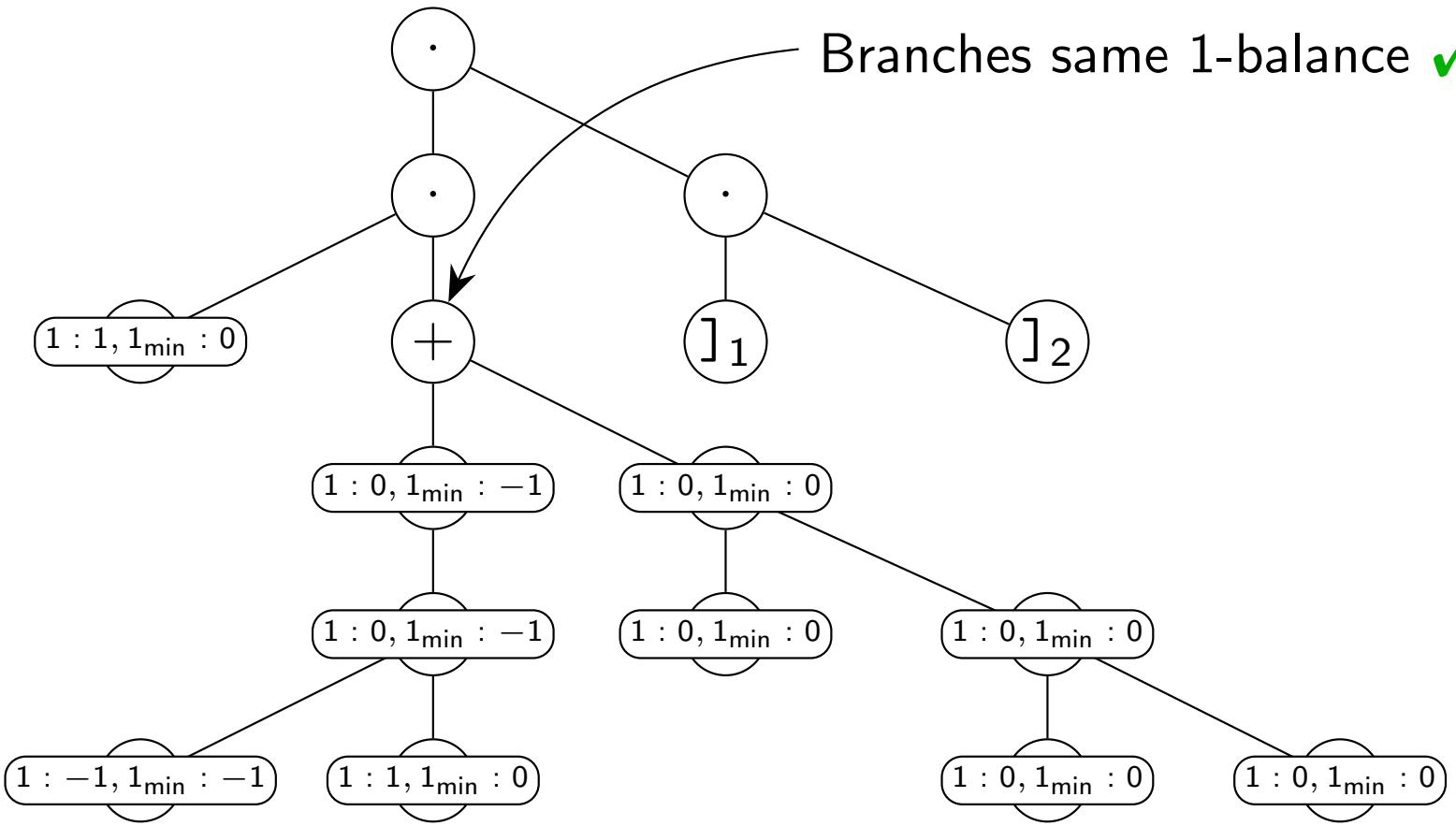
Checking the 1-balance



Recognising balanced regular languages

$e = [_1((]_1[_1)^* + [_2]_2[_2)]_1]_2$

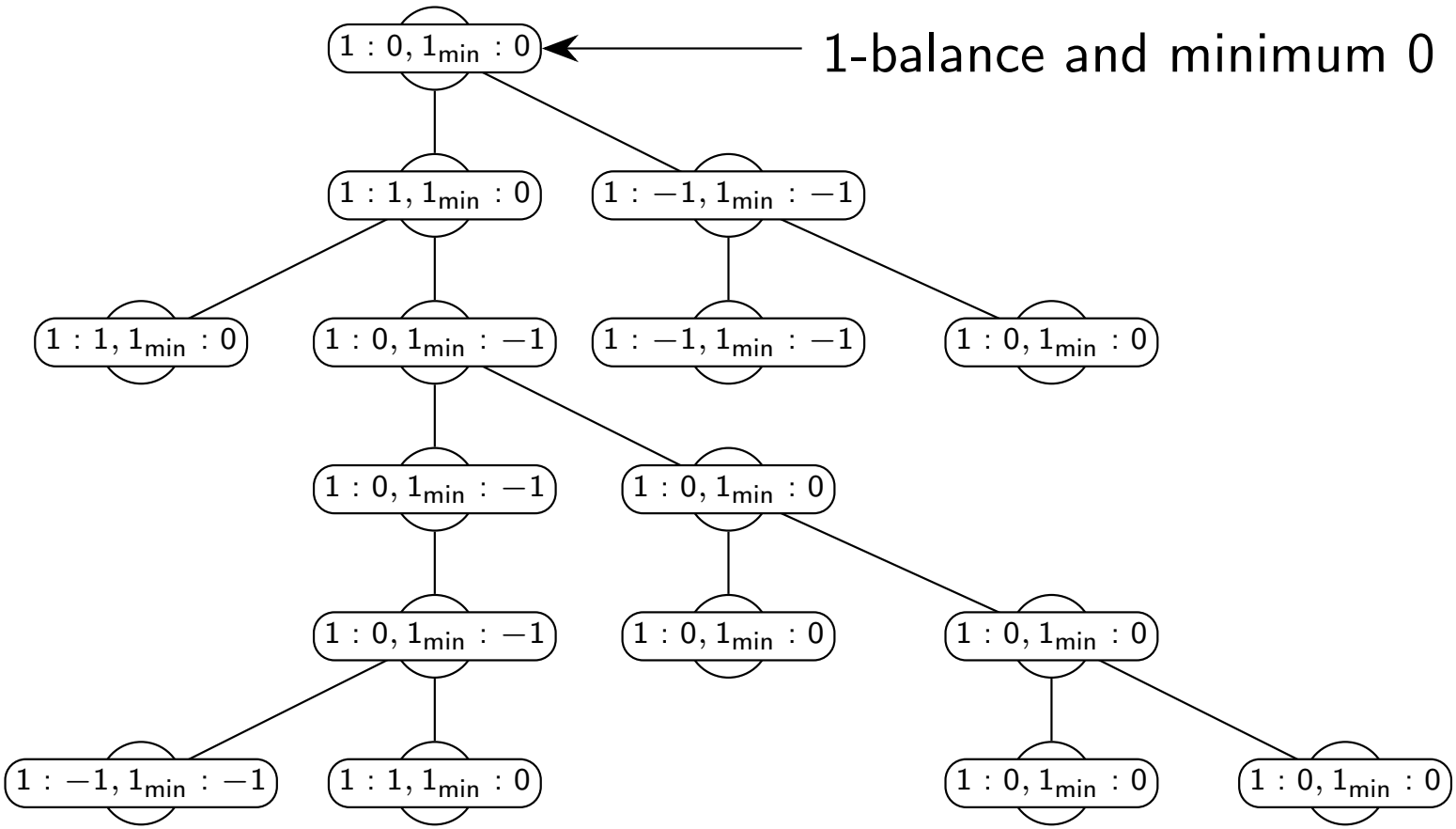
Checking the 1-balance



Recognising balanced regular languages

$e = [_1((]_1[_1)^* + [_2]_2[_2)]_1]_2$

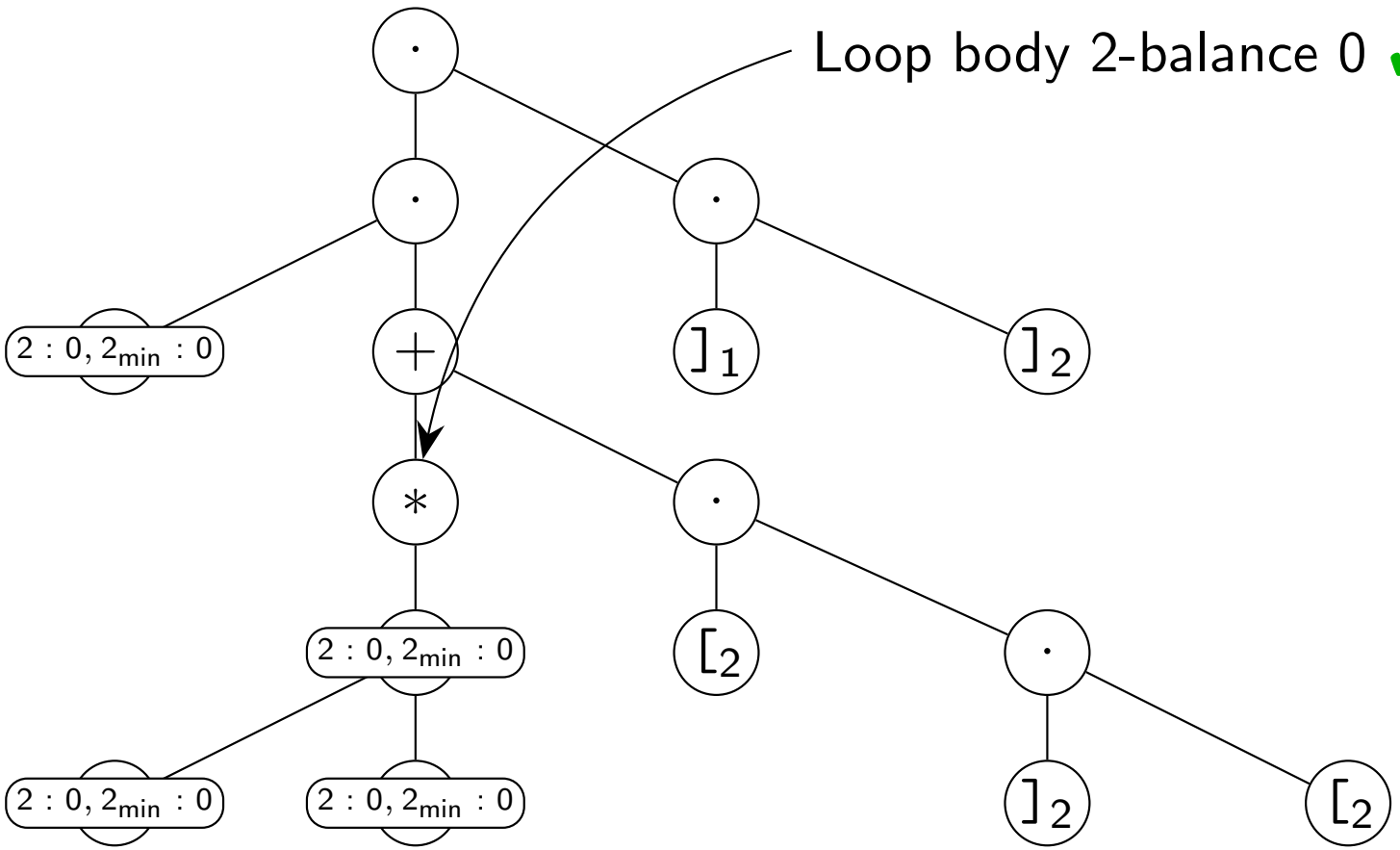
Checking the 1-balance



Recognising balanced regular languages

$e = [_1((]_1[_1)^* + [_2]_2[_2)]_1]_2$

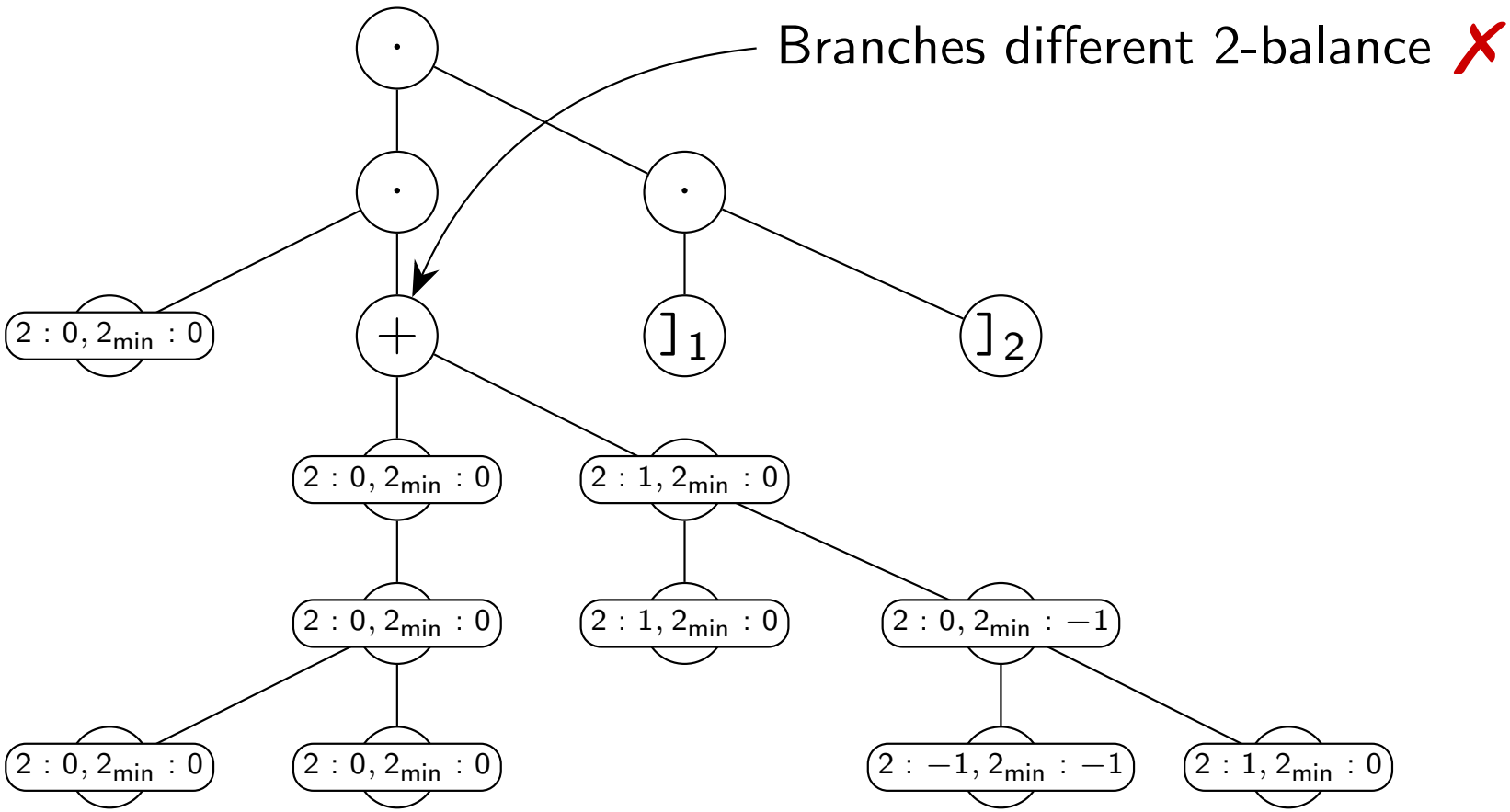
Checking the 2-balance



Recognising balanced regular languages

$e = [_1((]_1[_1)^* + [_2]_2[_2)]_1]_2$

Checking the 2-balance



Recognising balanced regular languages

Theorem

A regular expression is balanced if and only if its i-balance and minimum i-balance equal 0 for every i.

Expressing balanced regular languages

The grammar $\mathbb{E}^{\sqcup}$:

$$
\begin{aligned}
e ::= \varnothing &\mid \lambda \mid [_1 \cdot]_1 \mid [_2 \cdot]_2 \mid \ldots \\
&\mid e_1 + e_2 \mid e_1 \cdot e_2 \mid e^* \\
&\mid \sqcup^1_\theta (e_1) \mid \sqcup^2_\theta (e_1, e_2) \mid \ldots
\end{aligned}
$$

$$
\begin{aligned}
\theta ::= \varnothing &\mid \lambda \mid 1 \mid 2 \mid \ldots \\
&\mid \theta_1 + \theta_2 \mid \theta_1 \cdot \theta_2 \mid \theta^*
\end{aligned}
$$

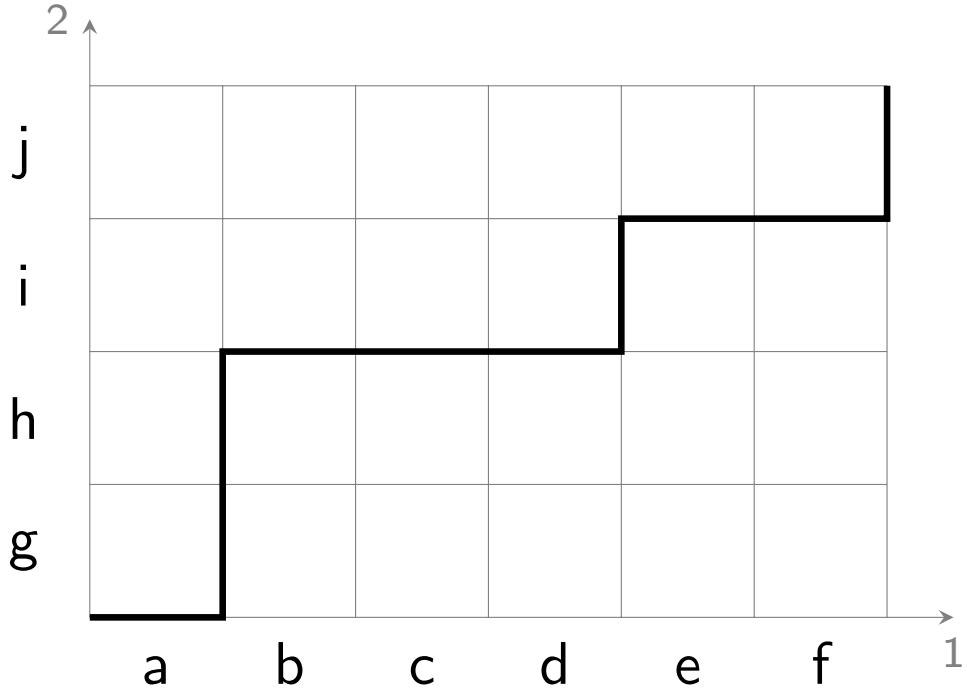
Expressing balanced regular languages

Mateescu et al.: “Shuffle on trajectories” (1998)

Trajectory

$\sqcup\!\sqcup^{2}_{1221112112}(\mathsf{abcdef}, \mathsf{ghij})$

$= \mathsf{aghbcdiefj}$



- Only defined if the trajectory fits the operands
- Generalises to languages and expressions

Expressing balanced regular languages

Soundness proof:

- Construct finite automaton for shuffle on trajectories operator
- Generalisation of Mateescu et al.

Expressing balanced regular languages

Completeness proof:

1. Regular expression in disjunctive normal form
2. Recursively rewrite subexpressions into shuffles of specific balanced and unbalanced ‘factors’
3. Number of unbalanced i-factors correlates with i-balance and minimum i-balance

Expressing balanced regular languages

Balanced factors

$$\bigcirc_i^k = ([_i]_i)^k([_i]_i)^* \to \square \qquad (\bigcirc_i^k)^* \to \blacksquare$$

Unbalanced factors

$$\oplus_i^k = \bigcirc_i^k [_i \to \square$$

$$\ominus_i^k =]_i \bigcirc_i^k \to \square$$

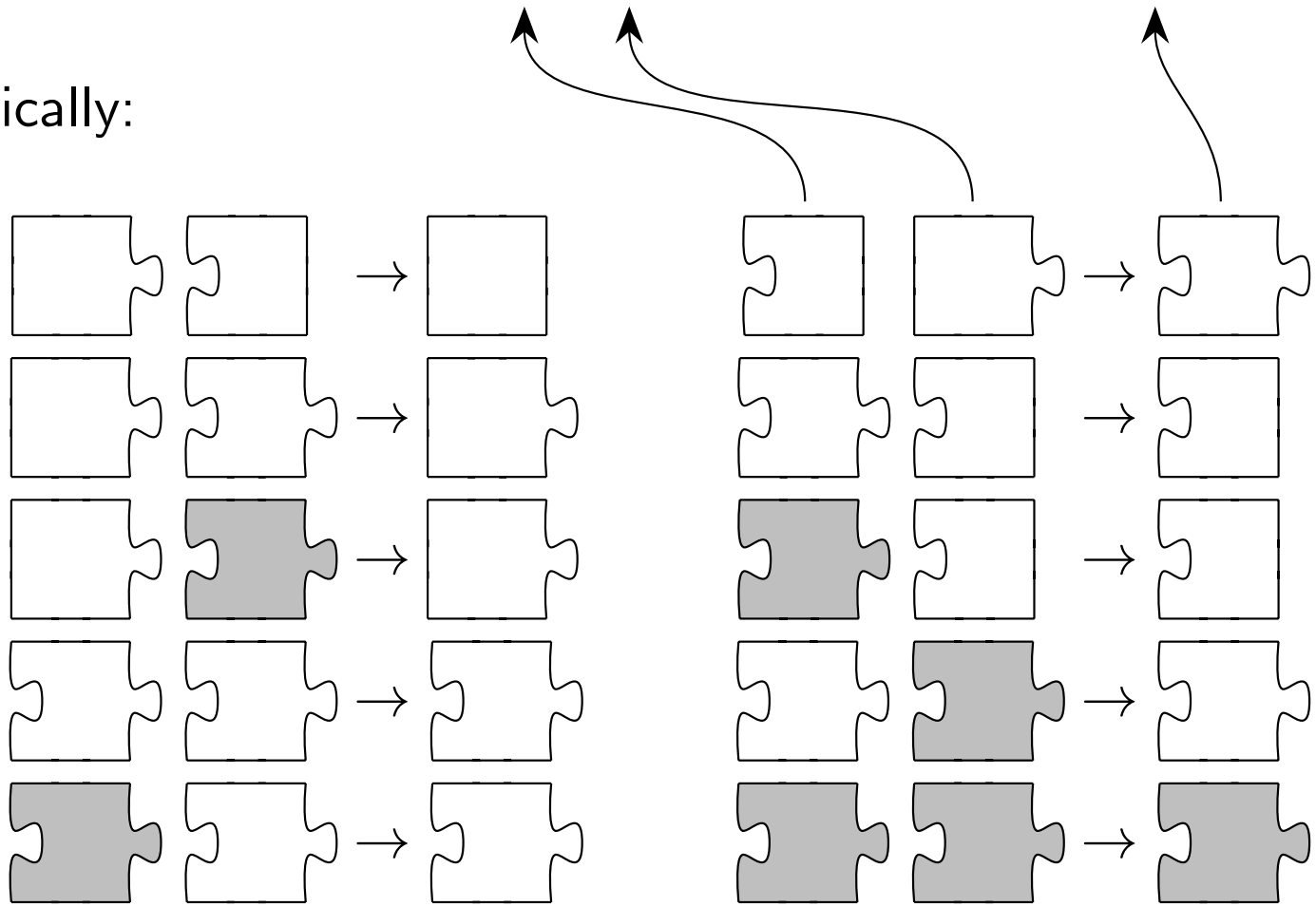
$$\pm_i^k =]_i \bigcirc_i^k [_i \to \square \qquad (\pm_i^k)^* \to \blacksquare$$

Expressing balanced regular languages

Lemma (Merge)

If [...] then $\sqcup_T(L_1, \ldots, L_{n-1}, L_n) = \sqcup_{T'}(L_1, \ldots, L_{n-1}L_n)$.

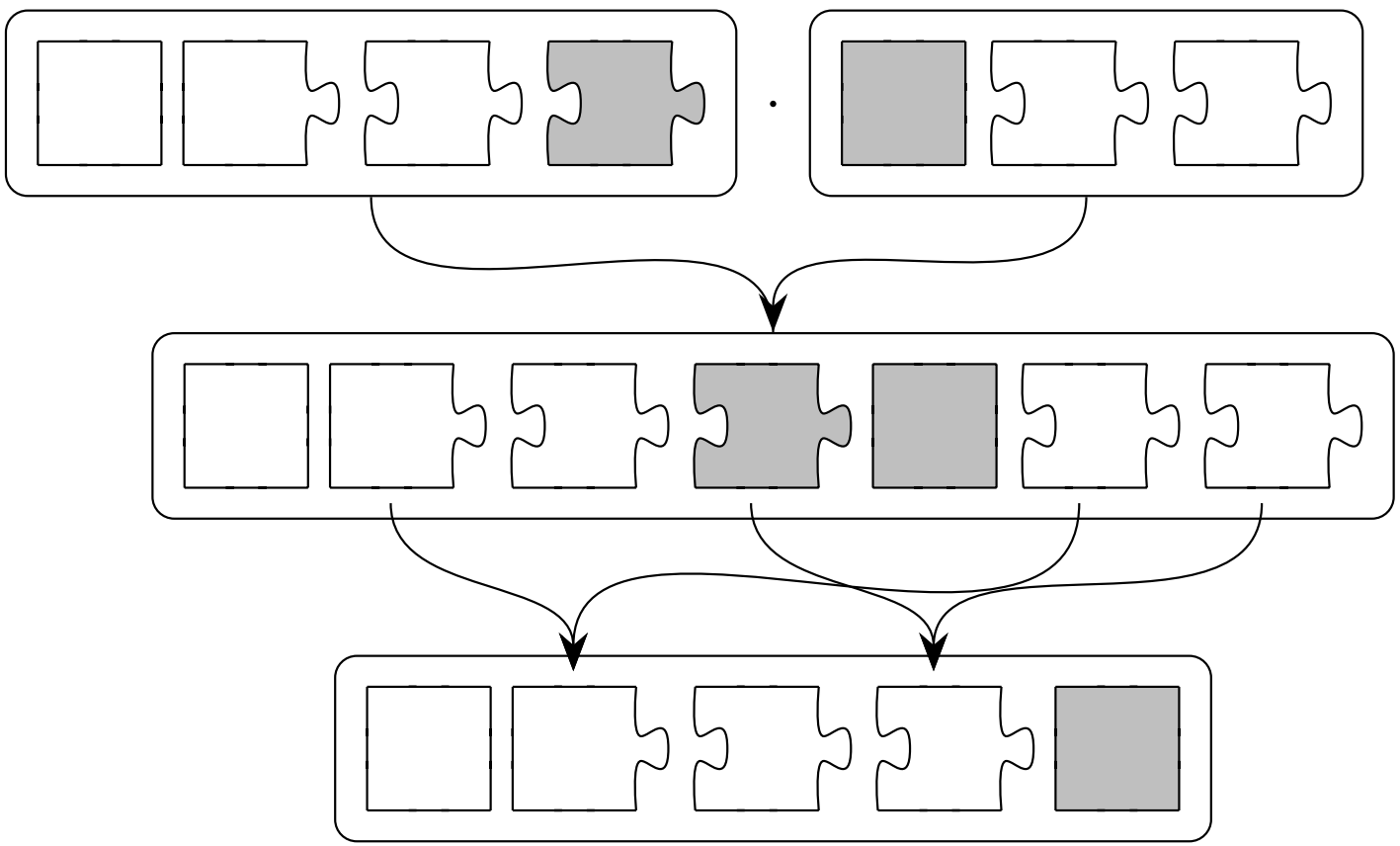
Specifically:



Expressing balanced regular languages

Lemma (Concatenation)

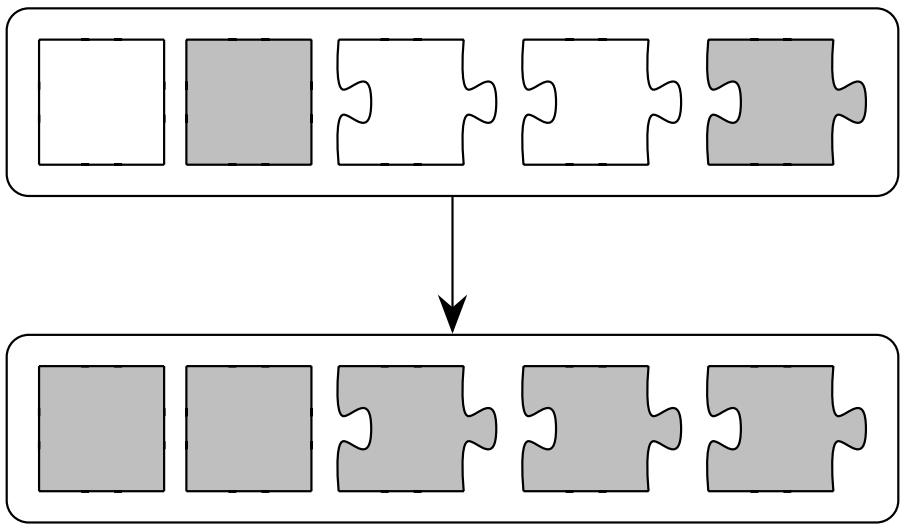
If [...] then $\sqcup_{T_1}(L_1, \ldots, L_n) \cdot \sqcup_{T_2}(L_{n+1}, \ldots, L_{n+m}) = \sqcup_{T}(L_1, \ldots, L_{n+m}) = \sqcup_{T'}(L_k, \ldots, L_\ell)$.



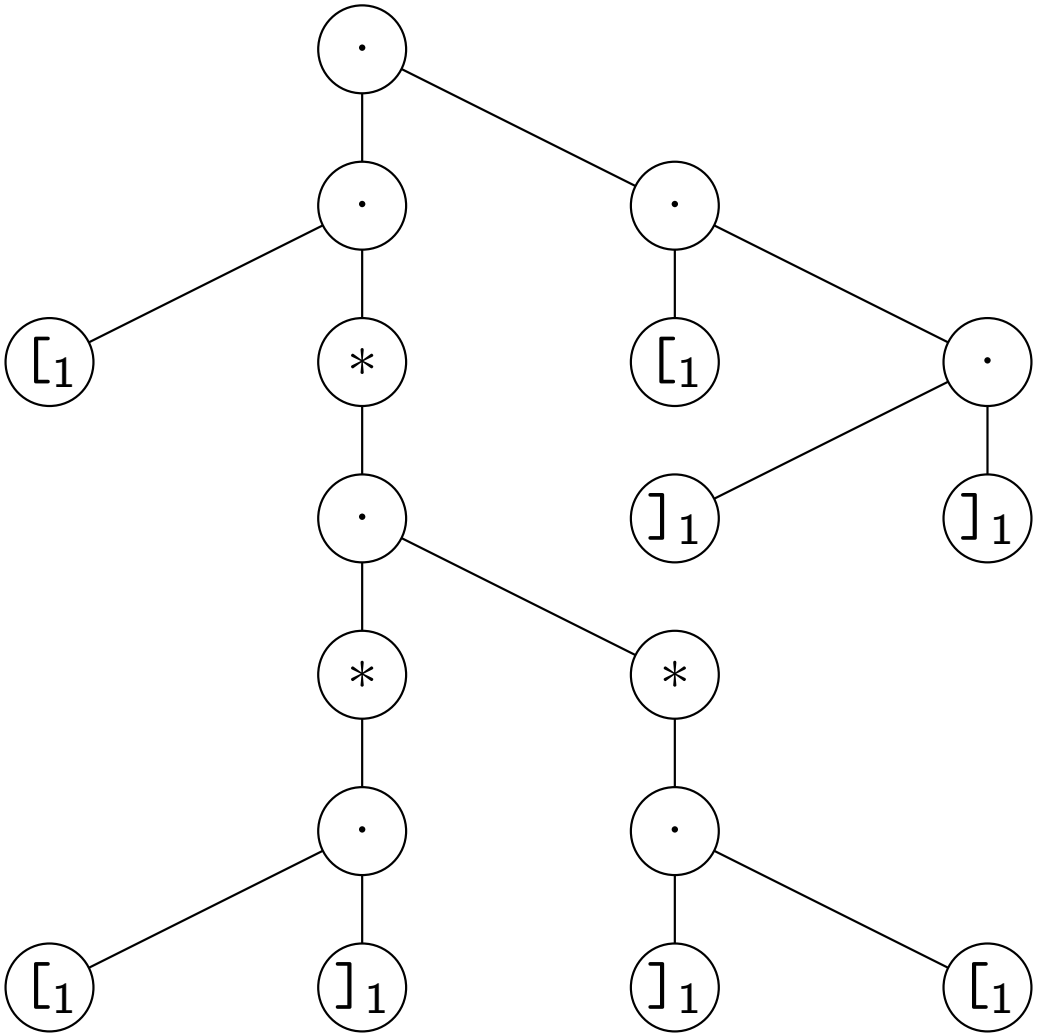
Expressing balanced regular languages

Lemma (Star)

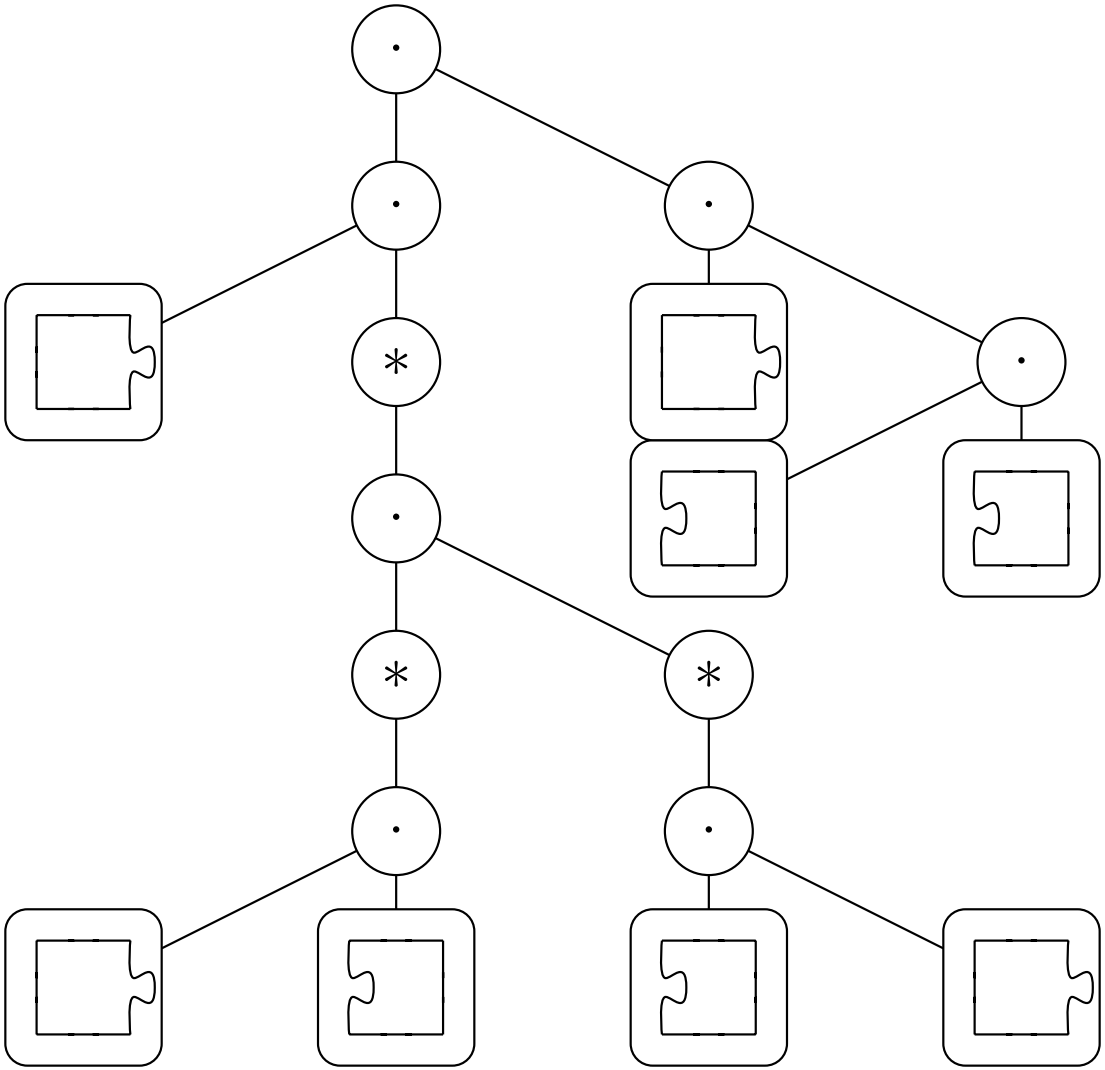
If [. . .] then $(\sqcup_T(L_1, \ldots, L_n))^* = \sqcup_{T^*}(L_1^*, \ldots, L_n^*)$.



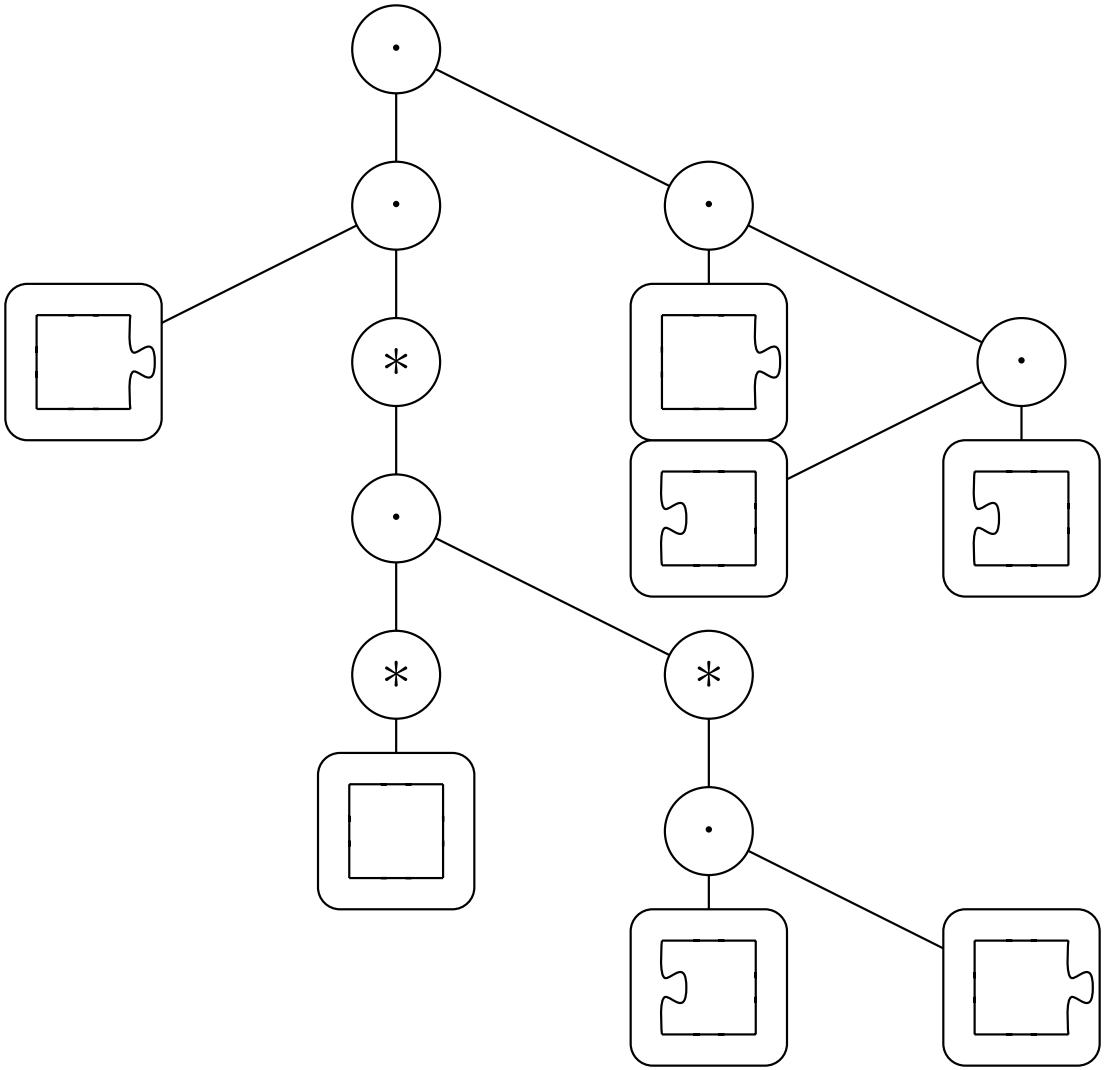
Expressing balanced regular languages



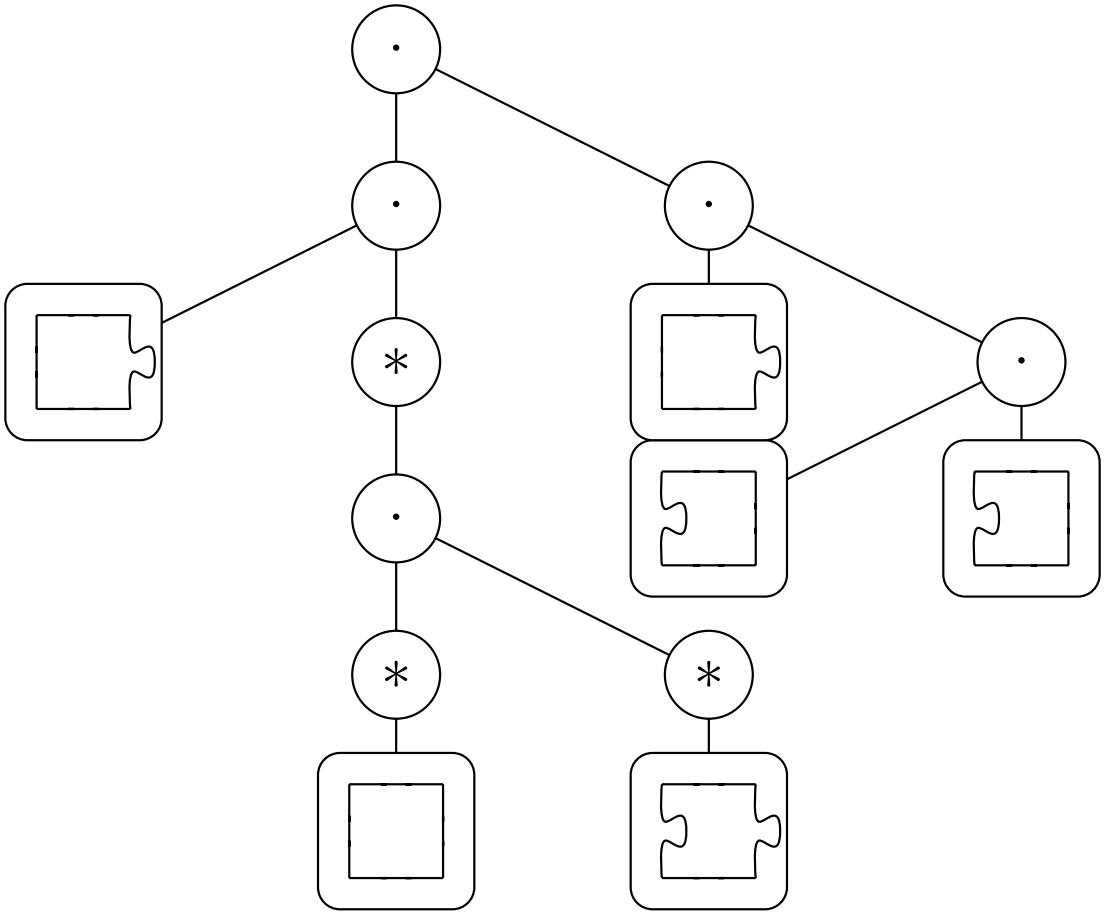
Expressing balanced regular languages



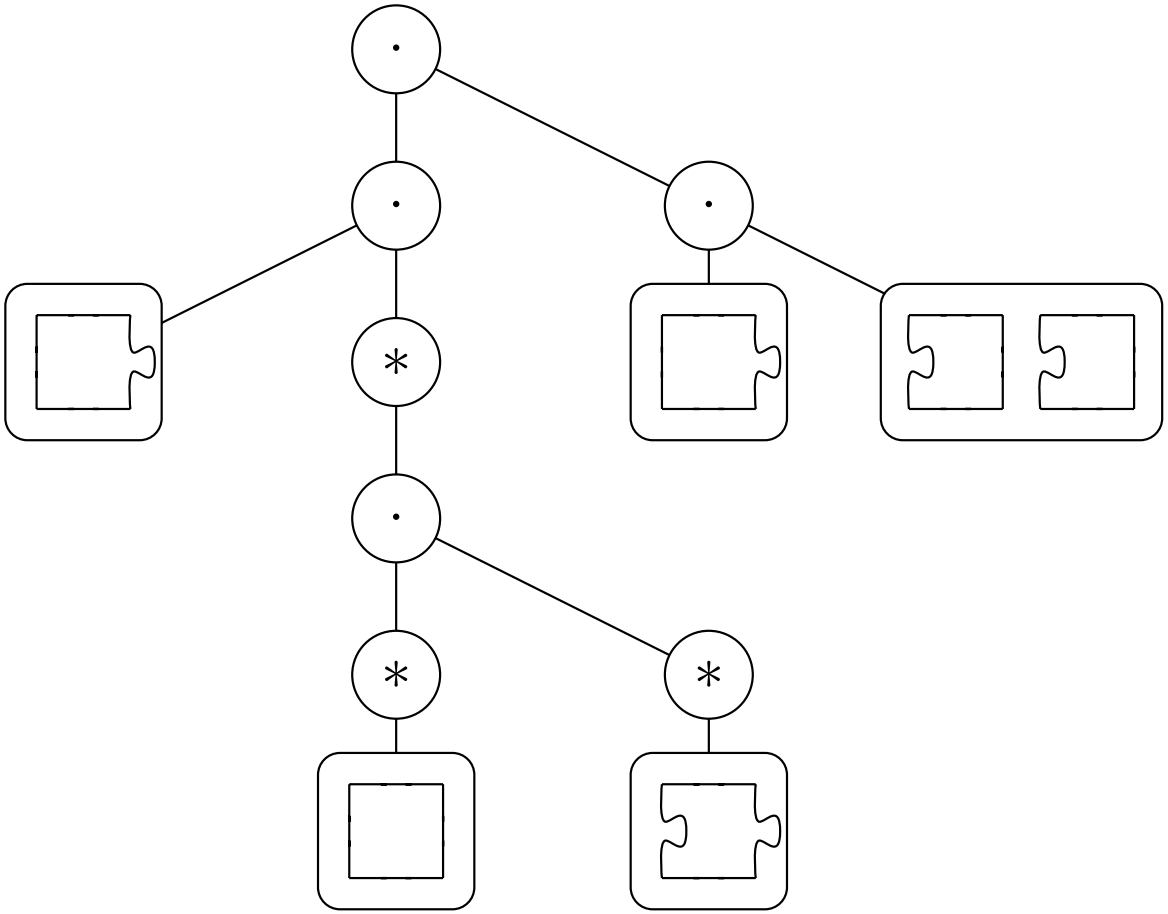
Expressing balanced regular languages



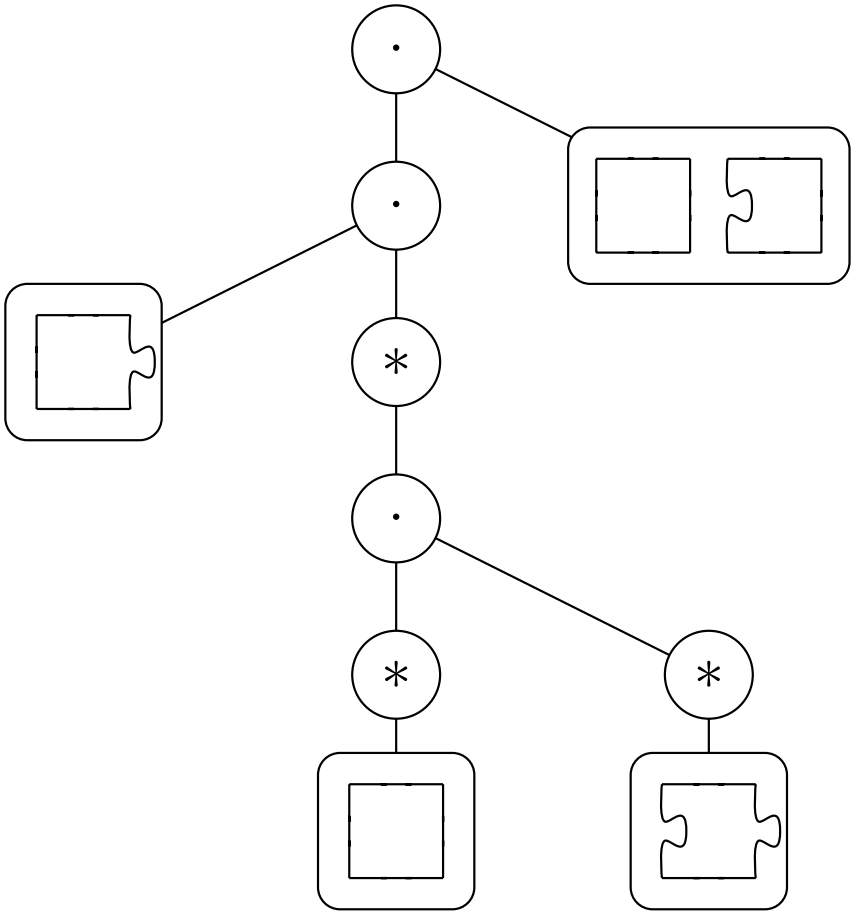
Expressing balanced regular languages



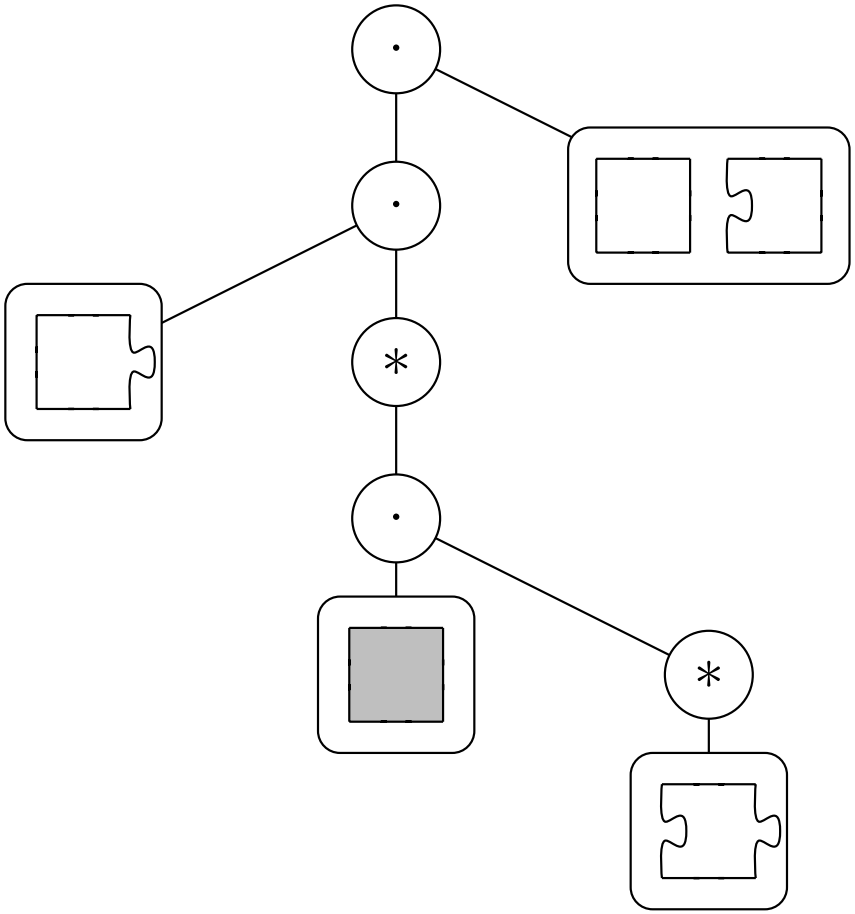
Expressing balanced regular languages



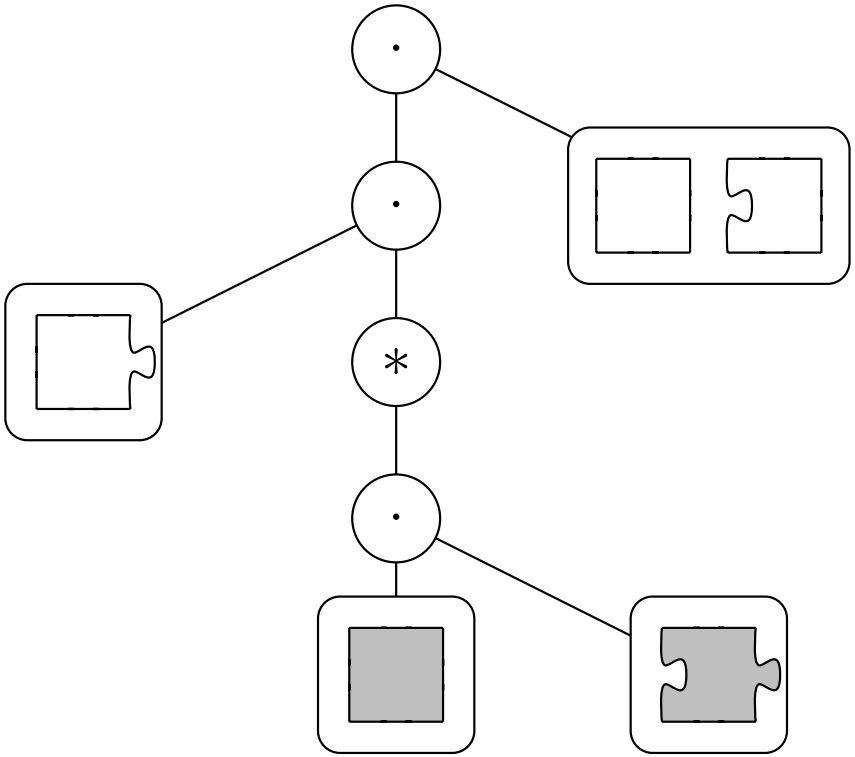
Expressing balanced regular languages



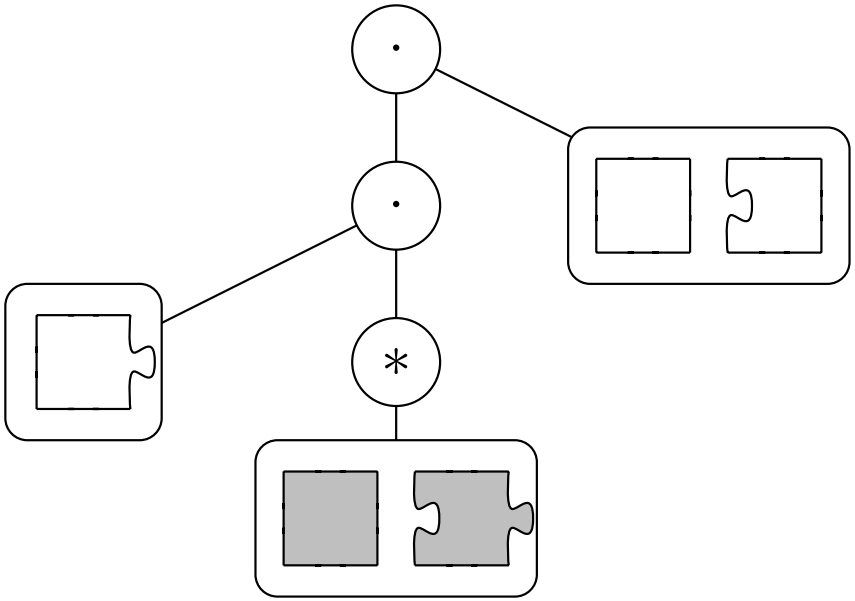
Expressing balanced regular languages



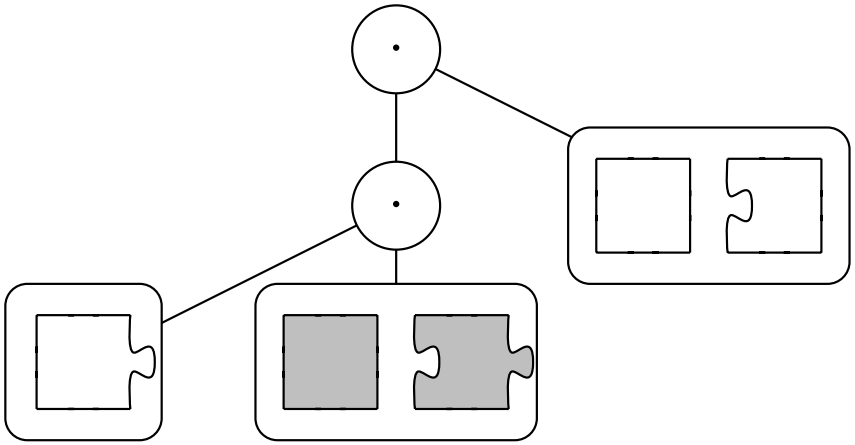
Expressing balanced regular languages



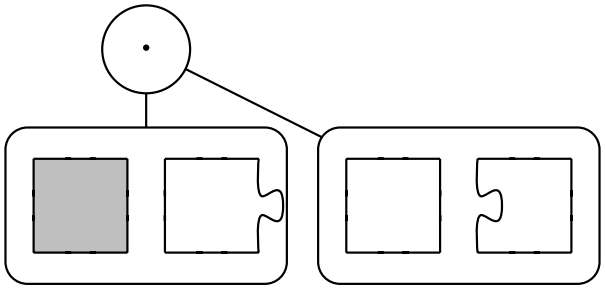
Expressing balanced regular languages



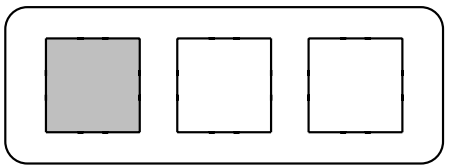
Expressing balanced regular languages



Expressing balanced regular languages



Expressing balanced regular languages



Expressing balanced regular languages

Theorem

The grammar $\mathbb{E}^{\sqcup}$ can express all and only all balanced regular languages.

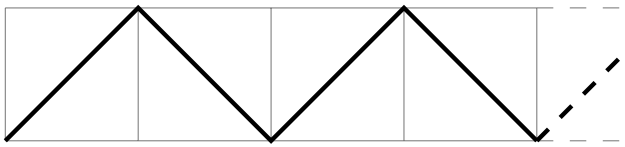
Balanced ω-regular languages

- Infinite words
- ω-regular expressions
- ω-automata (Büchi, Muller, . . .)

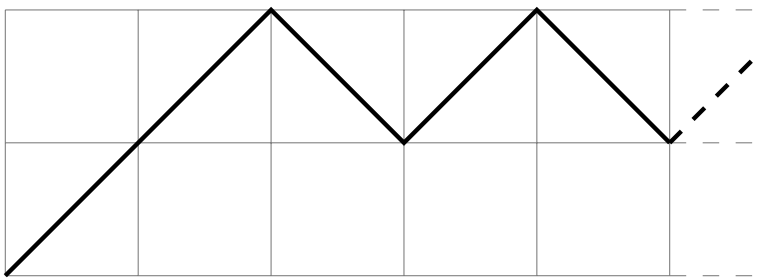
Balanced ω-regular languages

Balancedness $\rightarrow$ Boundedness

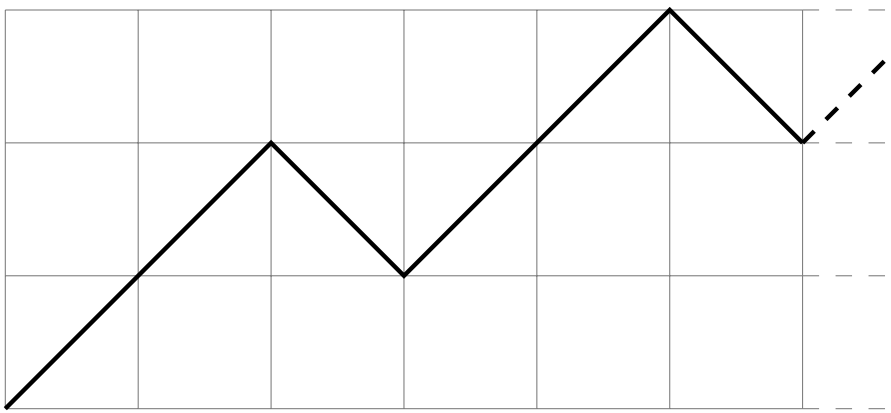
$([])^\omega$



$[([])^\omega$



$([[]])^\omega$



Balanced ω-regular languages

Theorem

An ω-regular expression is balanced if and only if its lower, upper and minimum i-balance equal 0 for every i.

Balanced ω-regular languages

The grammar $\Omega^{\sqcup\!\sqcup}$:

$$
\begin{aligned}
e &::= \varnothing \mid e + e \mid E \cdot e \mid E_+^\omega \mid \sqcup\!\sqcup_{T_\omega}(C, \ldots, C) && (\omega\text{-regular})\\
E &::= \varnothing \mid \lambda \mid P \mid E + E \mid E \cdot E \mid E^* \mid \sqcup\!\sqcup_{T}(E, \ldots, E) && (\text{regular})\\
E_+ &::= \varnothing \mid P \mid E_+ + E_+ \mid E \cdot E_+ \cdot E \mid \sqcup\!\sqcup_{T_+}(E, \ldots, E) && (\text{no } \lambda)\\
P &::= [_1 \cdot]_1 \mid [_2 \cdot]_2 \mid \ldots && (\text{parentheses})\\
C &::= e \mid E && (\omega\text{-shuffle operand})
\end{aligned}
$$

$$
\begin{aligned}
T &::= \varnothing \mid \lambda \mid 1 \mid 2 \mid \ldots \mid T + T \mid T \cdot T \mid T^* && (\text{trajectory})\\
T_+ &::= \varnothing \mid 1 \mid 2 \mid \ldots \mid T_+ + T_+ \mid T \cdot T_+ \cdot T && (\text{no } \lambda)\\
T_\omega &::= \varnothing \mid T_\omega + T_\omega \mid T \cdot T_\omega \mid T_+^\omega && (\omega\text{-trajectory})
\end{aligned}
$$

Balanced ω-regular languages

Theorem

The grammar $\Omega^{\sqcup\!\sqcup}$ can express all and only all balanced ω-regular languages.

Conclusions and future work

Conclusions

- Recognise balanced regular and ω-regular languages
- Expression grammars covering precisely balanced regular and ω-regular languages

Future work

- Context-free languages
- Finite number of operators

Expressing balanced regular languages

Closure of regular languages under shuffle on (regular) trajectories:

$$
\begin{aligned}
M = (&\Sigma_1 \cup \ldots \cup \Sigma_n, && \text{(alphabet)}\\
&Q_1 \times \ldots \times Q_n \times Q_t && \text{(states)}\\
&(q_0^1, \ldots, q_0^n, q_0^t) && \text{(initial state)}\\
&\delta && \text{(transition function)}\\
&F_1 \times \ldots \times F_n \times F_t) && \text{(final states)}
\end{aligned}
$$

where $\delta((q_1, \ldots, q_n, q_t), \sigma) =$
$\{(\delta_1(q_1, \sigma), \ldots, q_n, \delta_t(q_t, 1)), \ldots, (q_1, \ldots, \delta_n(q_n, \sigma), \delta_t(q_t, n))\}$.

Balanced ω-regular languages

Analysing ω-regular expressions

- $(\lambda + [_1)([_1]_1)^\omega$ is balanced, but $(\lambda + [_1)$ has no 1-balance

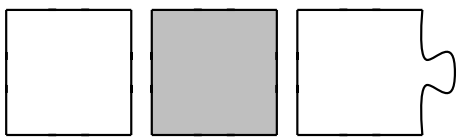
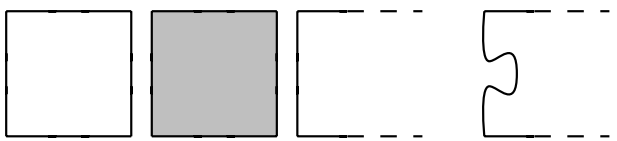
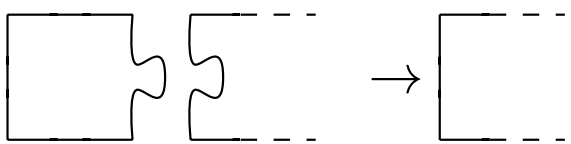
  $\rightarrow$ Split into lower and upper 1-balance

- $[_1([_1]_1)^\omega$ is balanced but $[_1([_2]_2)^\omega$ is not.

  $\rightarrow$ Keep track of guaranteed infinite occurrences of 1-brackets

Balanced ω-regular languages

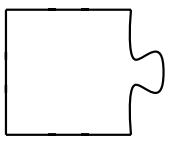
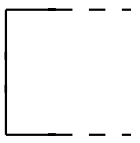
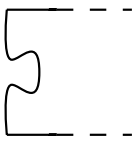
Expressing balanced ω-regular languages

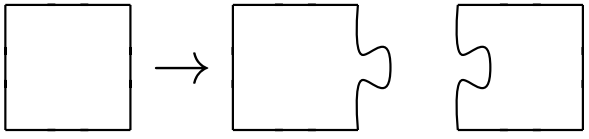
- Soundness: construct Muller automaton for shuffle operator
- Completeness: broadly as for regular languages
  - Normal form $e_1 e_2^\omega + \ldots$
  - $e_1 \rightarrow$ 
  - $e_2 \rightarrow$ 
  - Match 

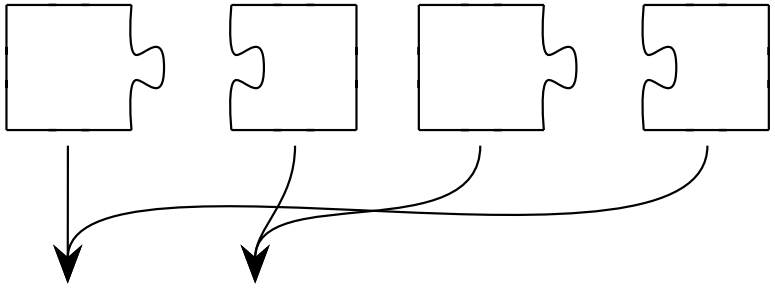
Balanced ω-regular languages

- $([_1]_1([_2]_2)^*)^\omega$ yields both words with finitely and infinitely many 2-brackets

  $\rightarrow$ Write as disjunction of shuffles

- $[_1([_1]_1)^\omega$ has factors  and  but no 

  $\rightarrow$ Reverse merge lemma, split inner 

  $\rightarrow e_2^\omega \equiv (e_2 e_2)^\omega \rightarrow$ 

  $\rightarrow$ Merge back into