Two-Way Visibly Pushdown Transducers

Luc Dartois [1], Emmanuel Filiot [1],
Pierre-Alain Reynier [2], Jean-Marc Talbot [2]

[1]Université Libre de Bruxelles

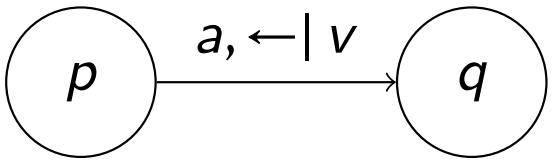
[2]Université d'Aix-Marseille

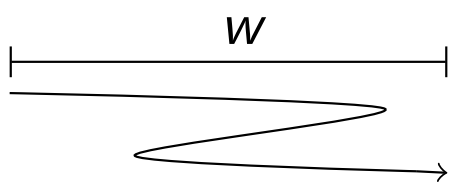
Highlights'16, September 9^{th} 2016

Models for word tranductions

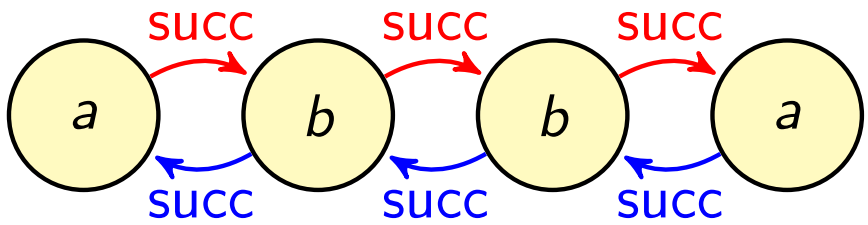
[Hoogeboom Engelfriet 01]

2-way Deterministic Transducers (2DFT) are as expressive as MSO Transductions (MSOT).





2DFT behavior



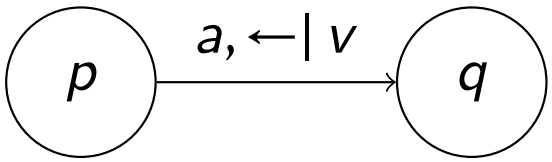
$$\phi_{succ}(x,y) \equiv succ(y,x)$$
$$\phi_{lab_a}(x) \equiv lab_a(x)$$

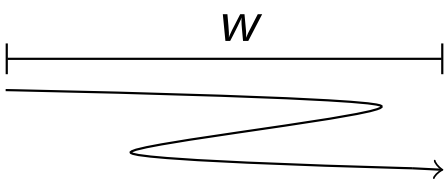
An MSO transduction

Models for word tranductions

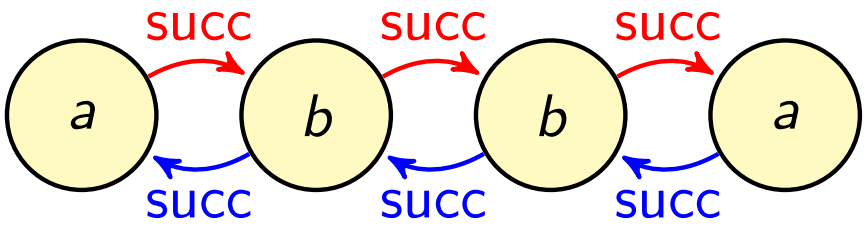
[Hoogeboom Engelfriet 01]

2-way Deterministic Transducers (2DFT) are as expressive as MSO Transductions (MSOT).





2DFT behavior



$$\phi_{succ}(x,y) \equiv succ(y,x)$$
$$\phi_{lab_a}(x) \equiv lab_a(x)$$

An MSO transduction

Goal of this talk

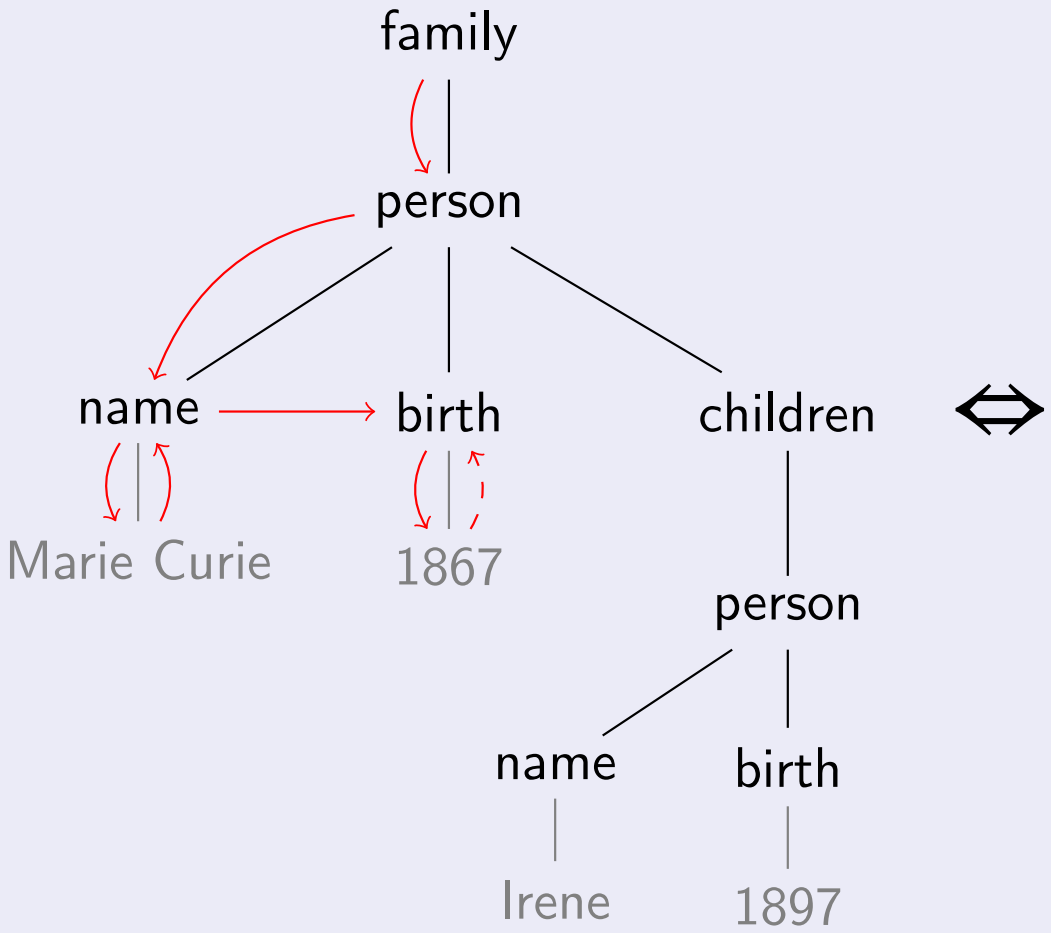
Extend this result to Nested Word to Word functions.

Well-nested words

Well-nested words are generated by the grammar $u, v \in \mathcal{N}(\Sigma) : \epsilon \mid u.v \mid cur$.

Encoding

Well-nested words ≡ linearizations of (unranked) trees



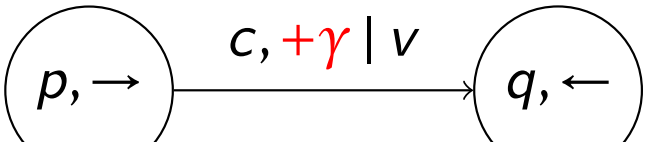
$\Leftrightarrow$

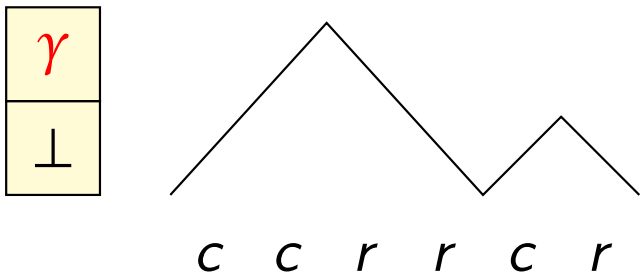
$<family>$
$<person>$
$<name>$ *MarieCurie* $<\backslash name>$
$<birth>$ 1867 $<\backslash birth>$
$<children>$
$<name>$ *Irene* $<\backslash name>$
$<birth>$ 1897 $<\backslash birth>$
$<\backslash children>$
$<\backslash person>$
$<\backslash family>$

Main result

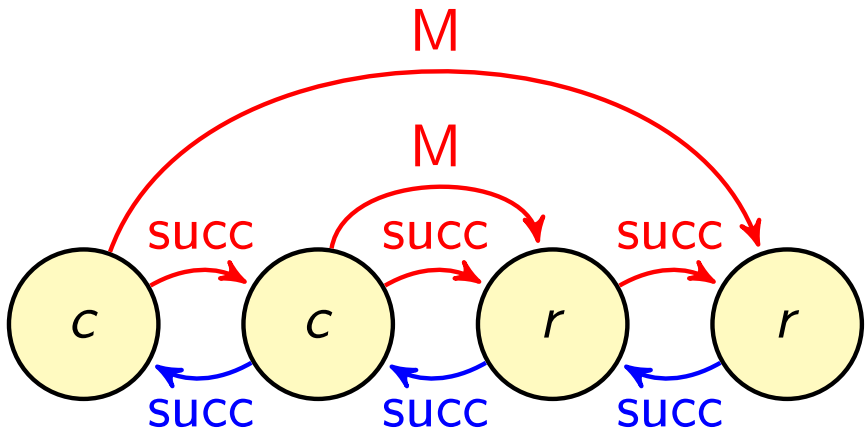
[D., Filiot, Reynier, Talbot 16]

Deterministic 2-way Visibly Pushdown Transducers (2VPT) are as expressive as MSO[nw2w] Transductions.



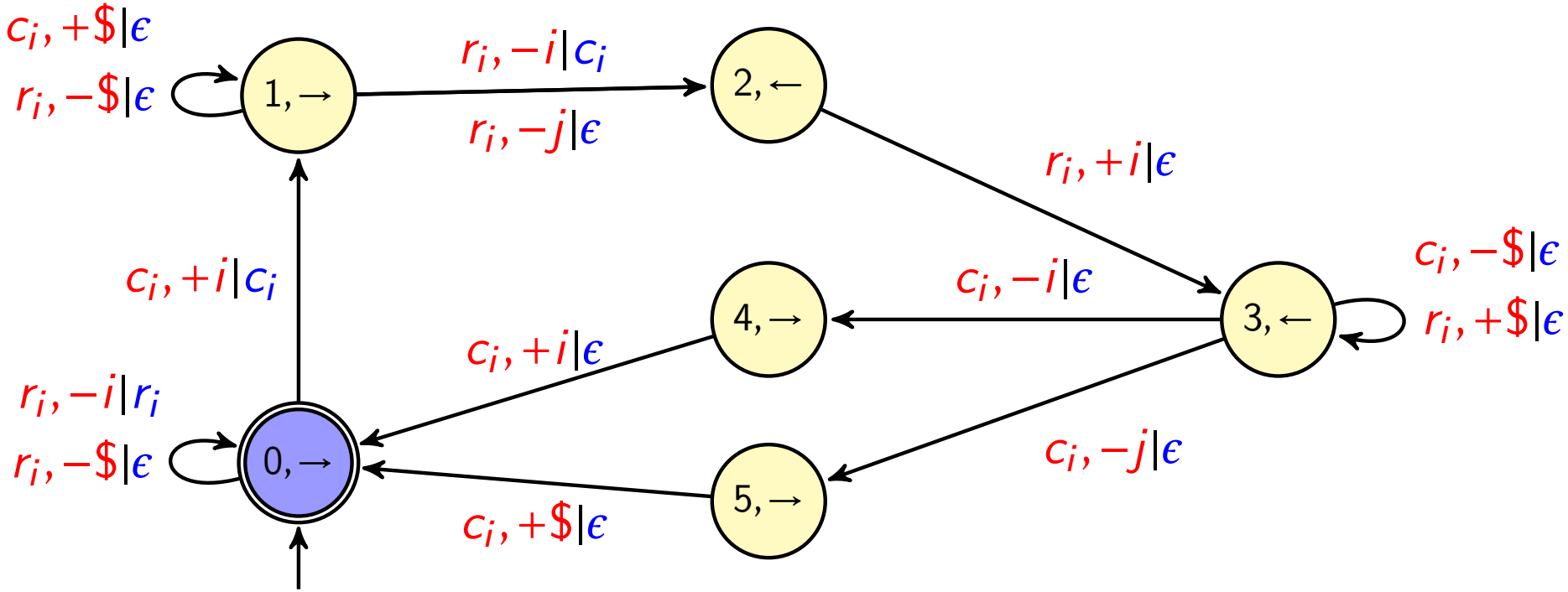


stack & stack height



$$\phi_{succ}(x,y) \equiv succ(y,x)$$
$$\phi_{lab_a}(x) \equiv lab_a(x)$$

Example: Deleting unequal matchings



Input Tape: $\vdash$ c_1 c_2 r_3 r_1 c_4 r_4 $\dashv$

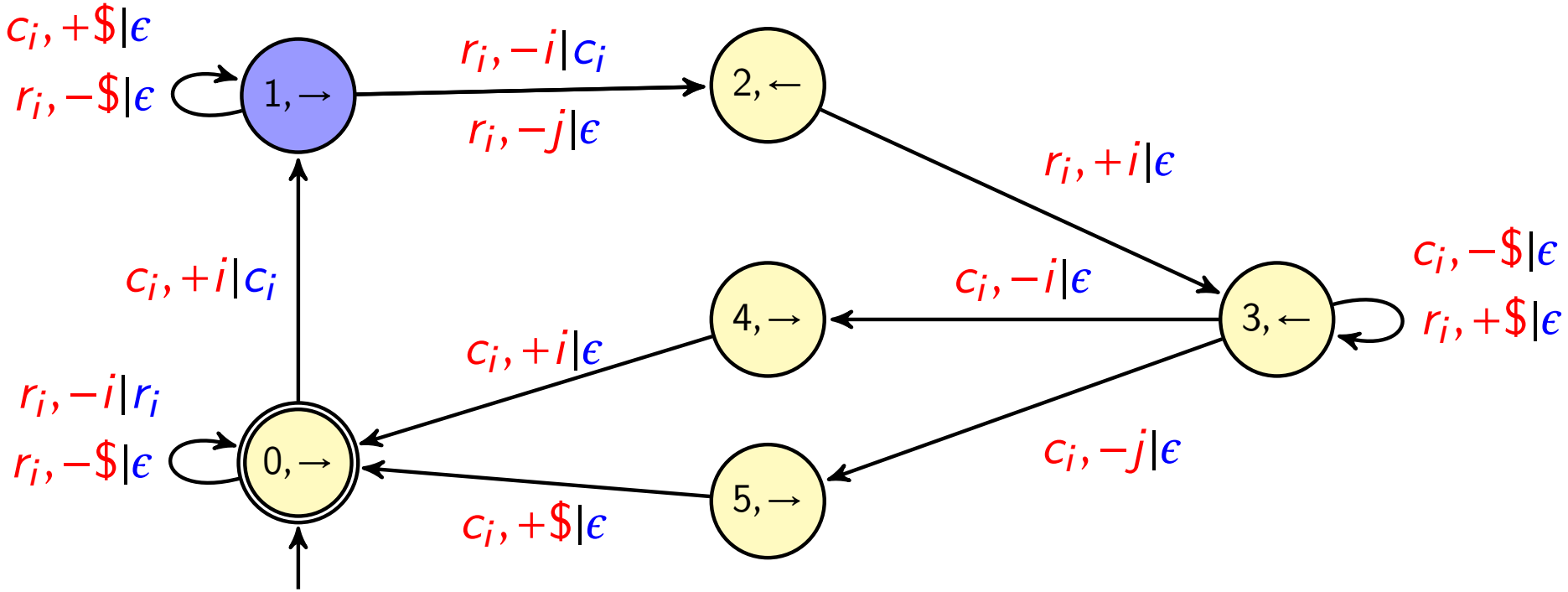
$(0, \rightarrow)$

$\perp$

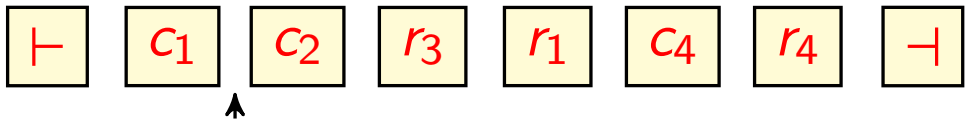
Output Tape:

stack

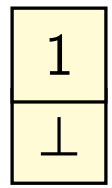
Example: Deleting unequal matchings



Input Tape:



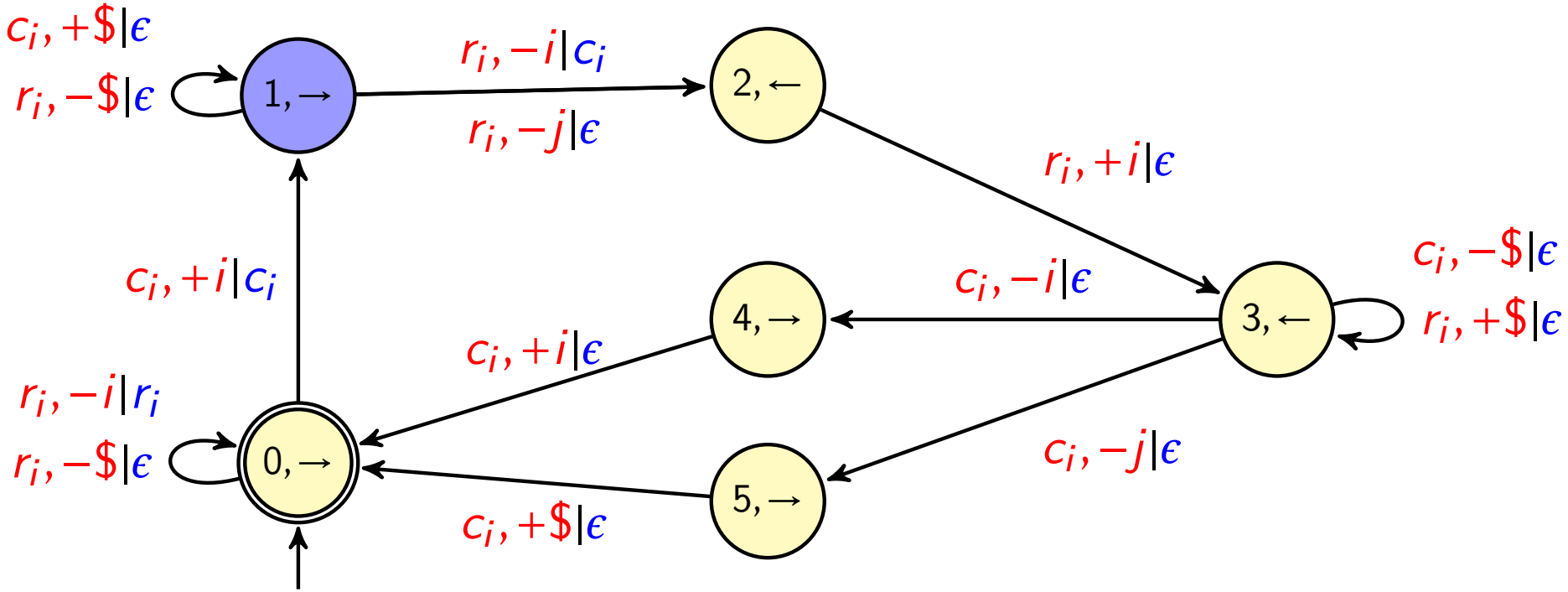
$(1, \rightarrow)$



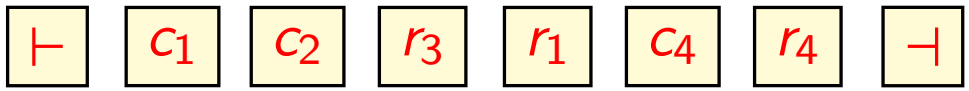
Output Tape:

stack

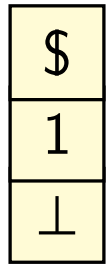
Example: Deleting unequal matchings



Input Tape:



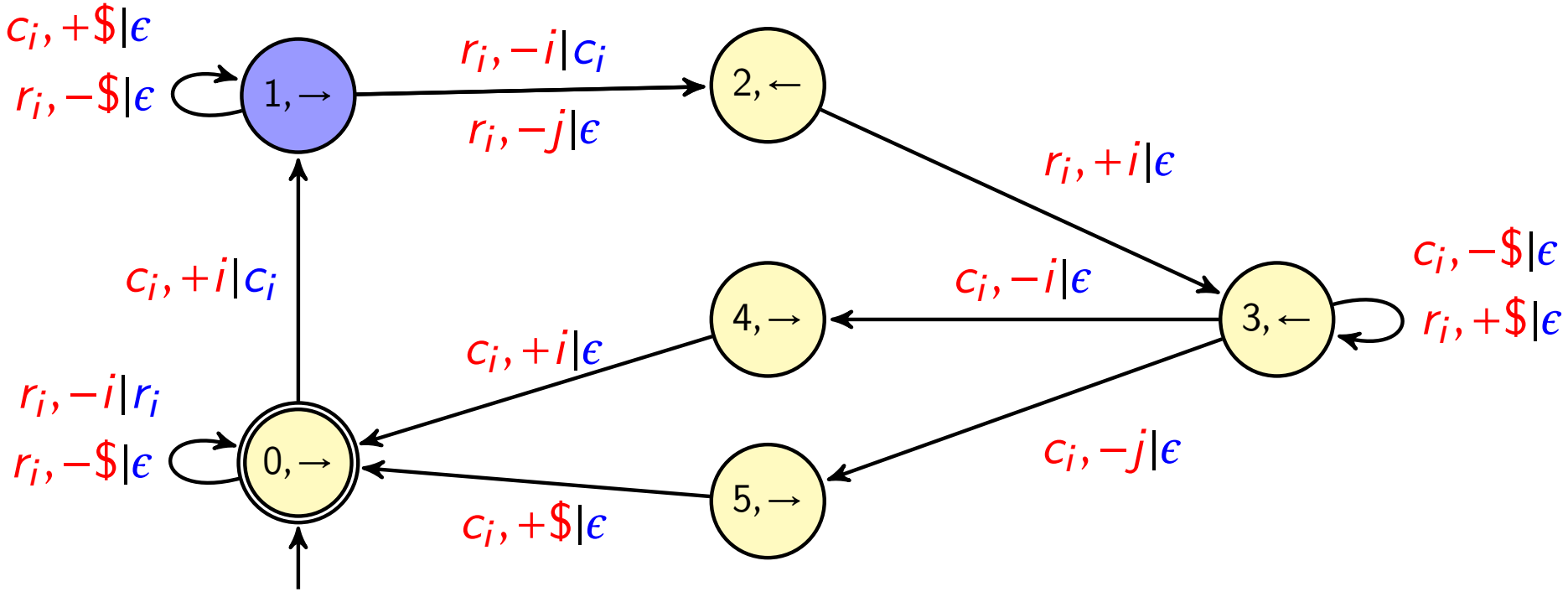
$(1, \rightarrow)$



Output Tape:

stack

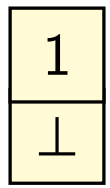
Example: Deleting unequal matchings



Input Tape: ⊢ c_1 c_2 r_3 r_1 c_4 r_4 ⊣

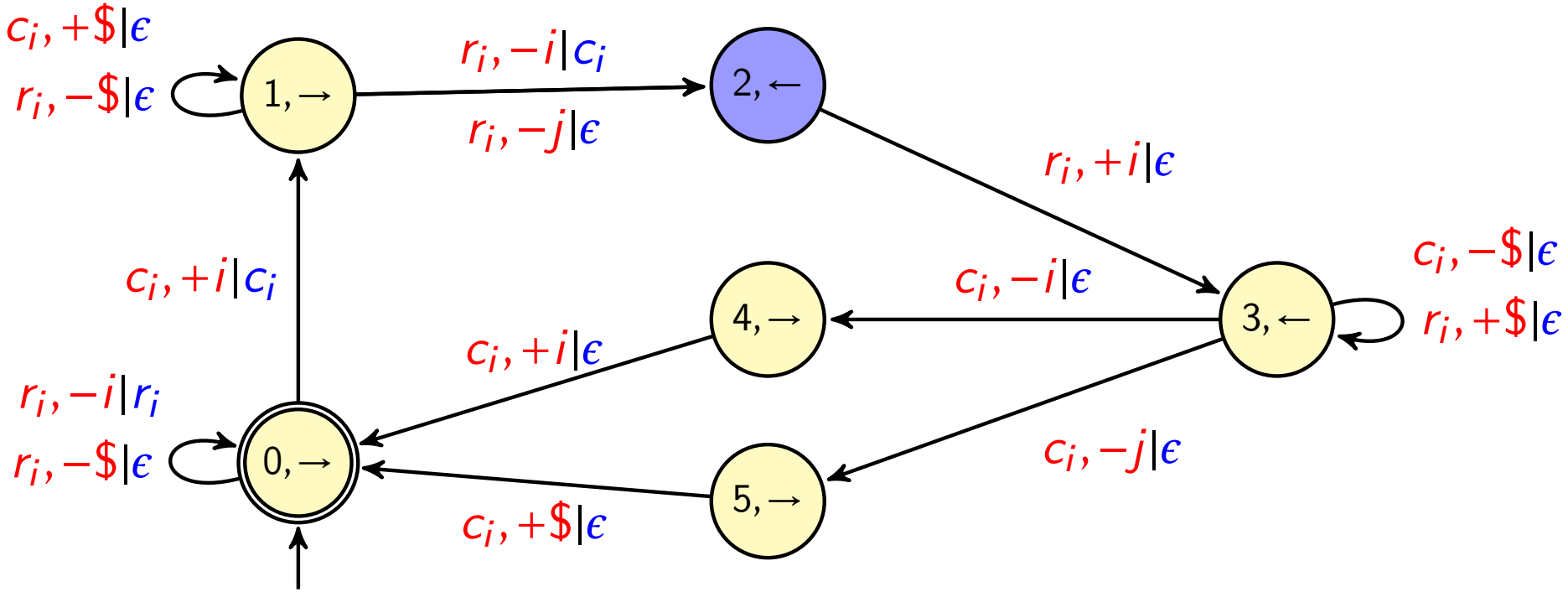
$(1, \rightarrow)$

Output Tape:



stack

Example: Deleting unequal matchings



Input Tape: $\vdash$ c_1 c_2 r_3 r_1 c_4 r_4 $\dashv$

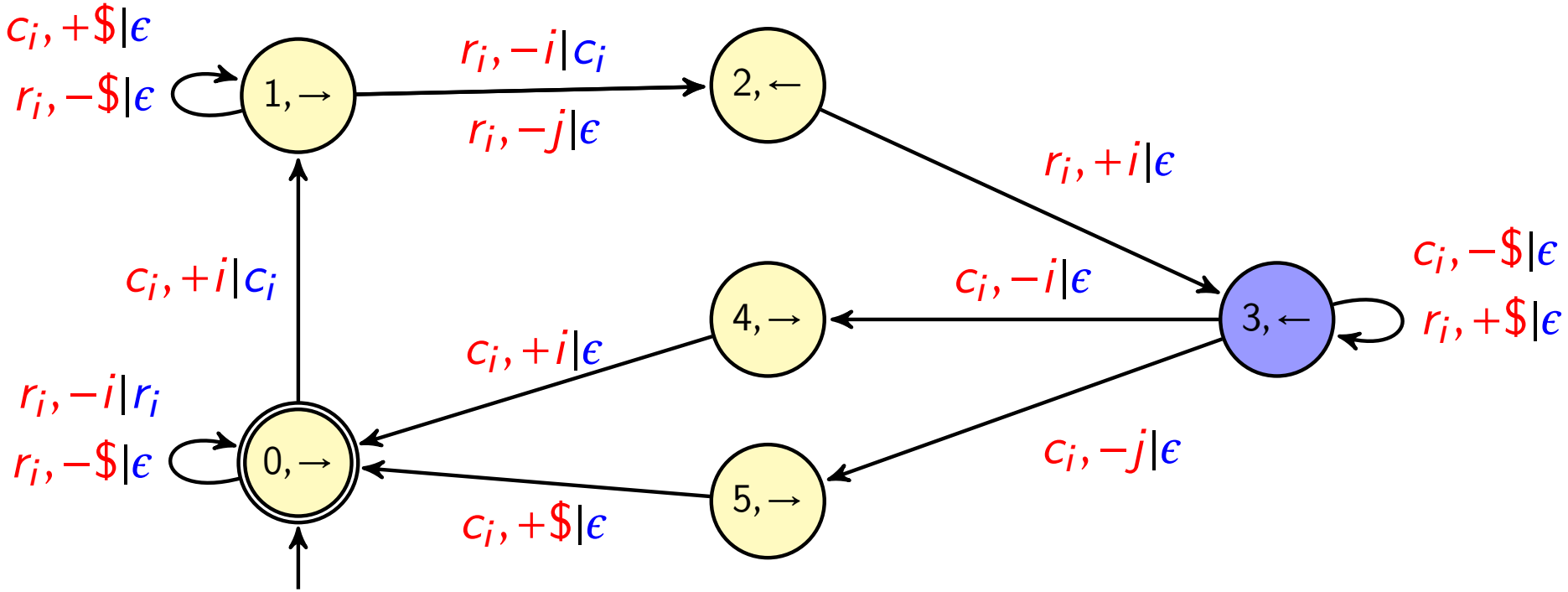
$(2, \leftarrow)$

$\bot$

stack

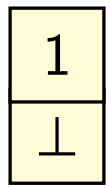
Output Tape: c_1

Example: Deleting unequal matchings



Input Tape: $\vdash$ c_1 c_2 r_3 r_1 c_4 r_4 $\dashv$

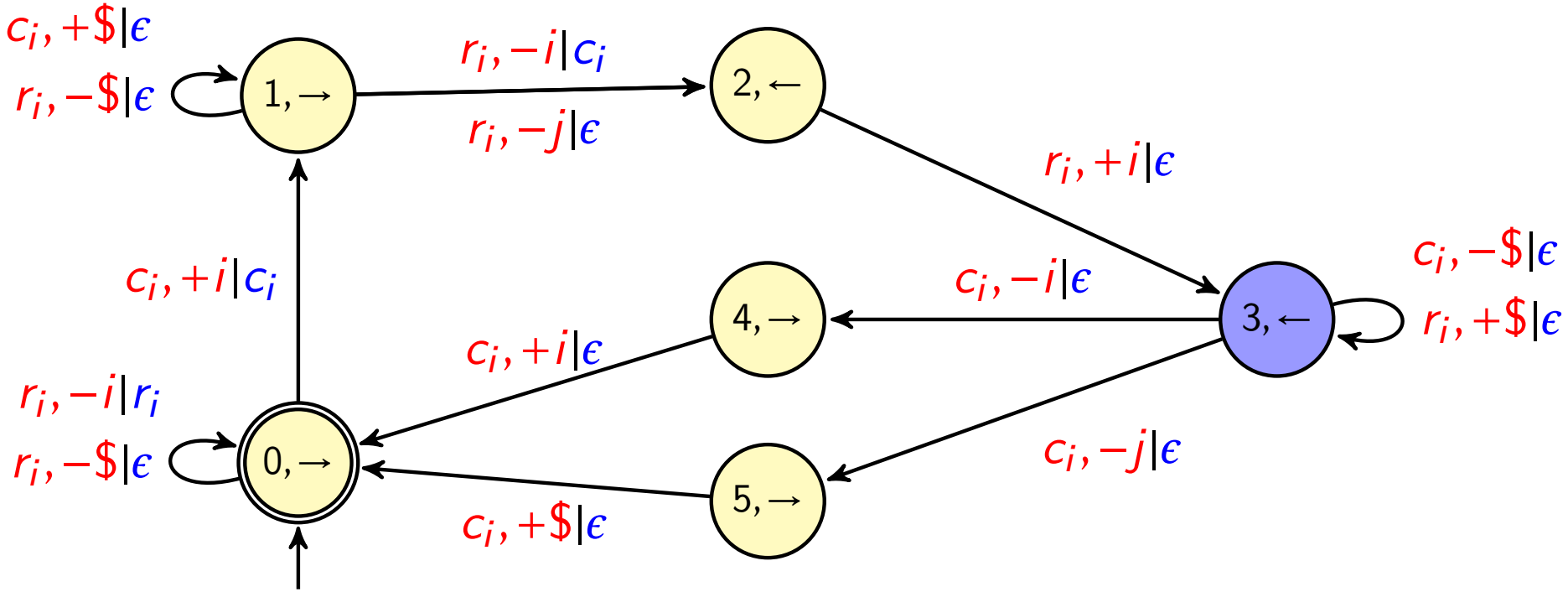
$(3, \leftarrow)$



stack

Output Tape: c_1

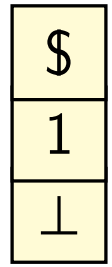
Example: Deleting unequal matchings



Input Tape: $\vdash$ c_1 c_2 r_3 r_1 c_4 r_4 $\dashv$

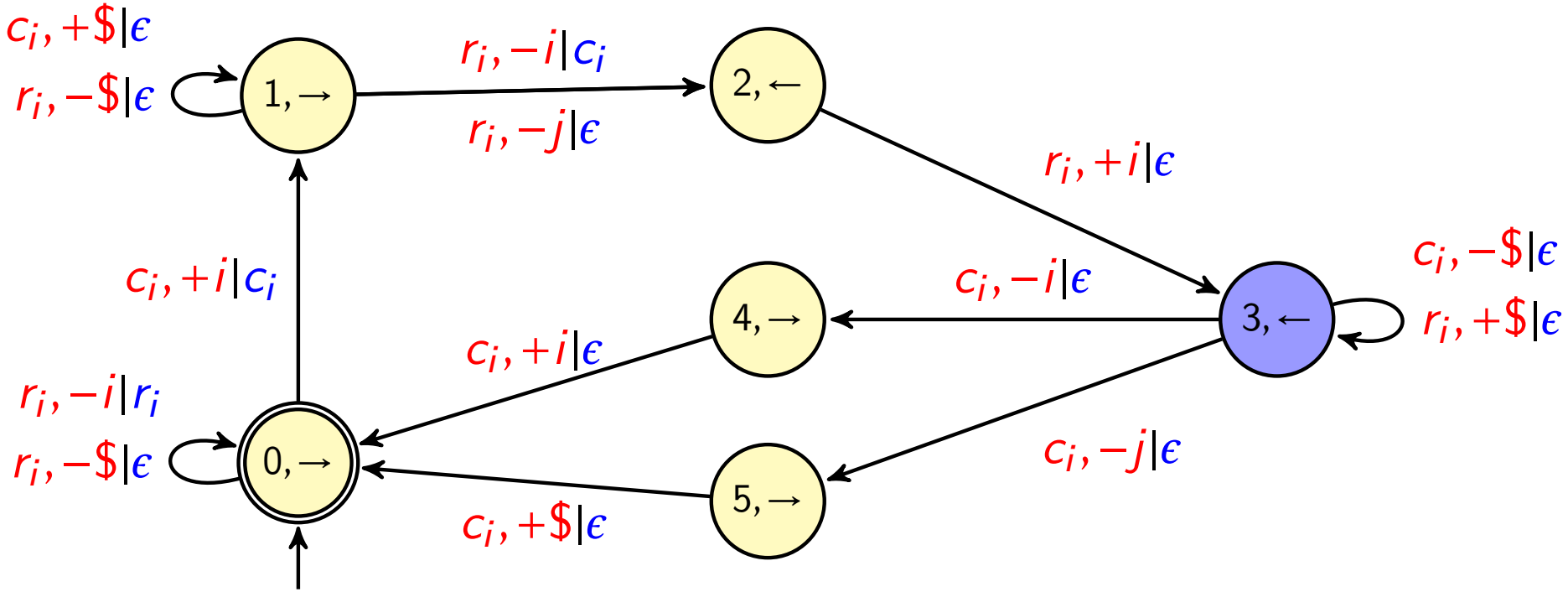
$(3, \leftarrow)$

Output Tape: c_1



stack

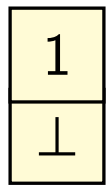
Example: Deleting unequal matchings



Input Tape: $\vdash$ c_1 c_2 r_3 r_1 c_4 r_4 $\dashv$

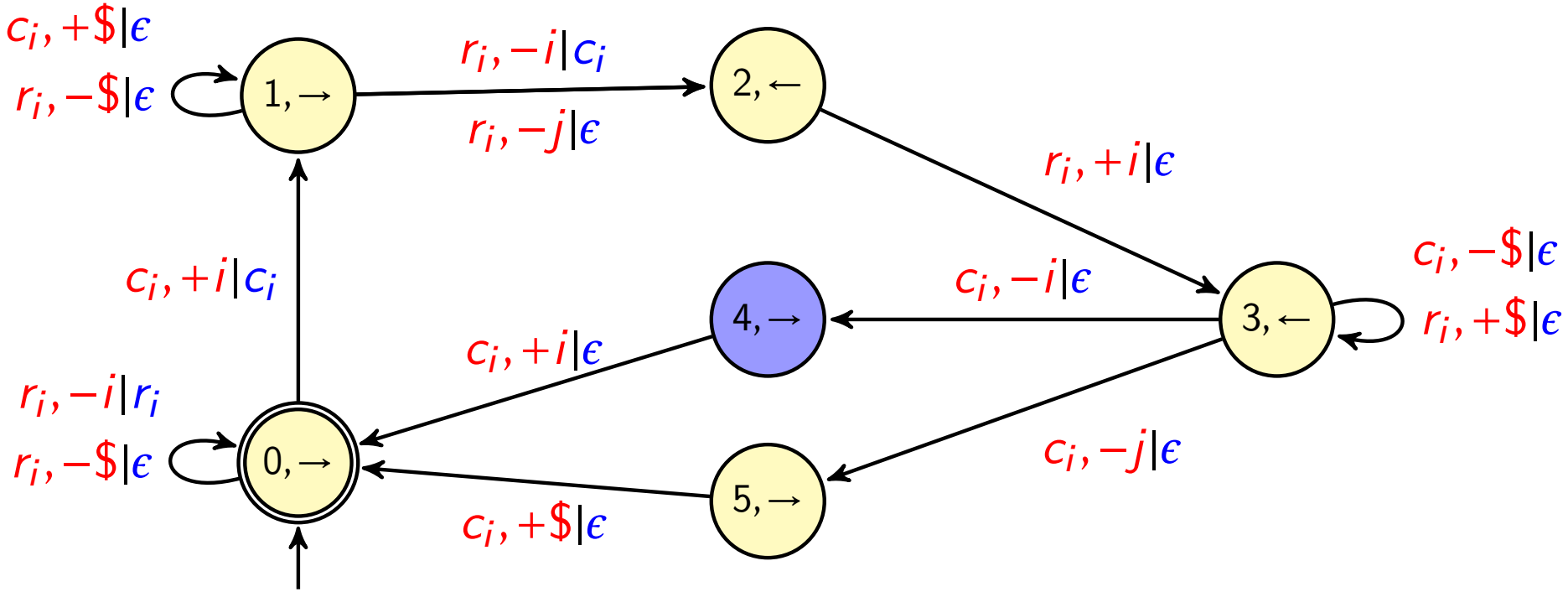
$(3, \leftarrow)$

Output Tape: c_1



stack

Example: Deleting unequal matchings



Input Tape: $\vdash$ c_1 c_2 r_3 r_1 c_4 r_4 $\dashv$

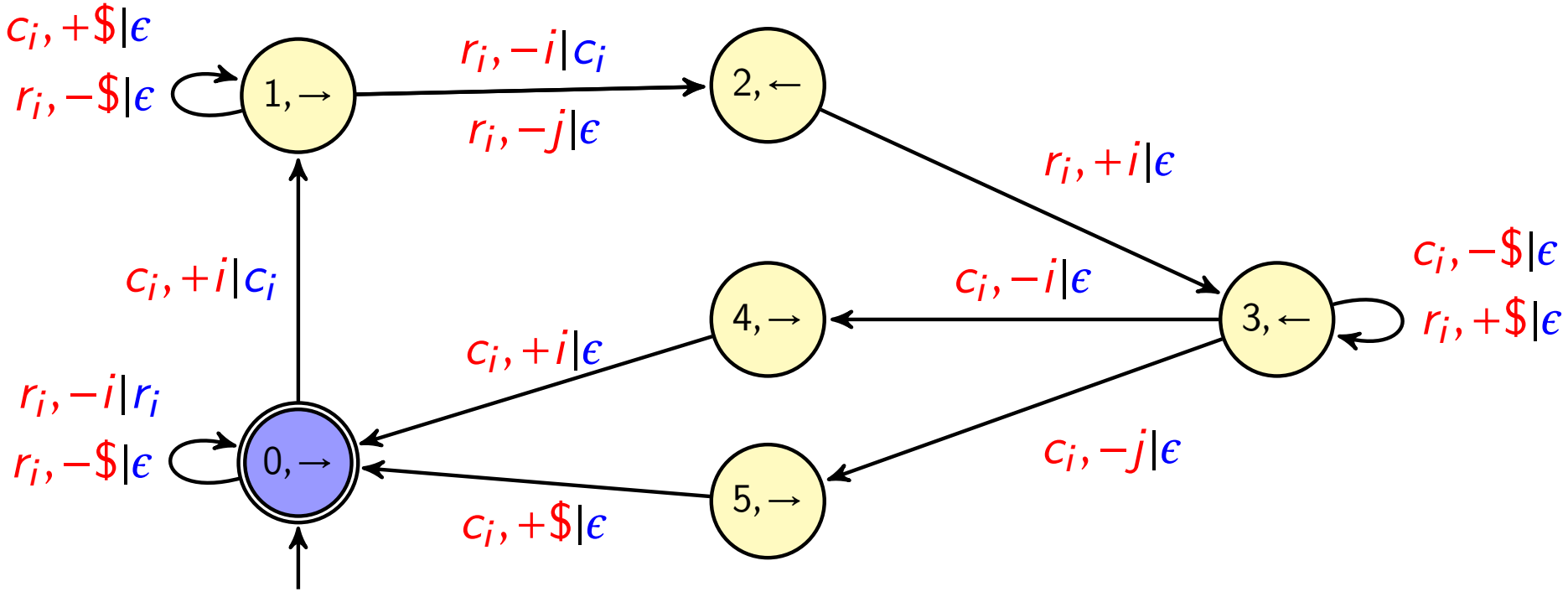
$(4, \rightarrow)$

Output Tape: c_1

$\perp$

stack

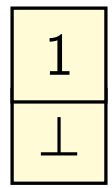
Example: Deleting unequal matchings



Input Tape: $\vdash$ c_1 c_2 r_3 r_1 c_4 r_4 $\dashv$

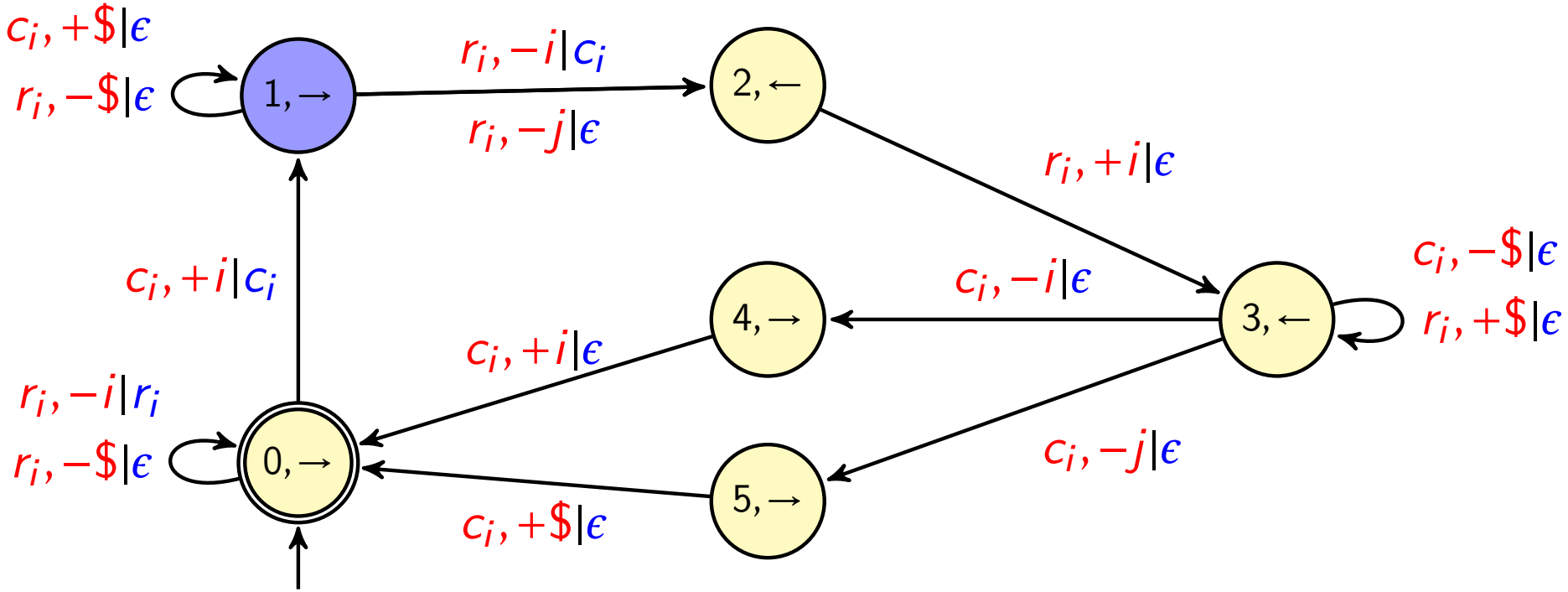
$(0, \rightarrow)$

Output Tape: c_1



stack

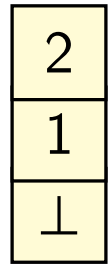
Example: Deleting unequal matchings



Input Tape: $\vdash$ c_1 c_2 r_3 r_1 c_4 r_4 $\dashv$

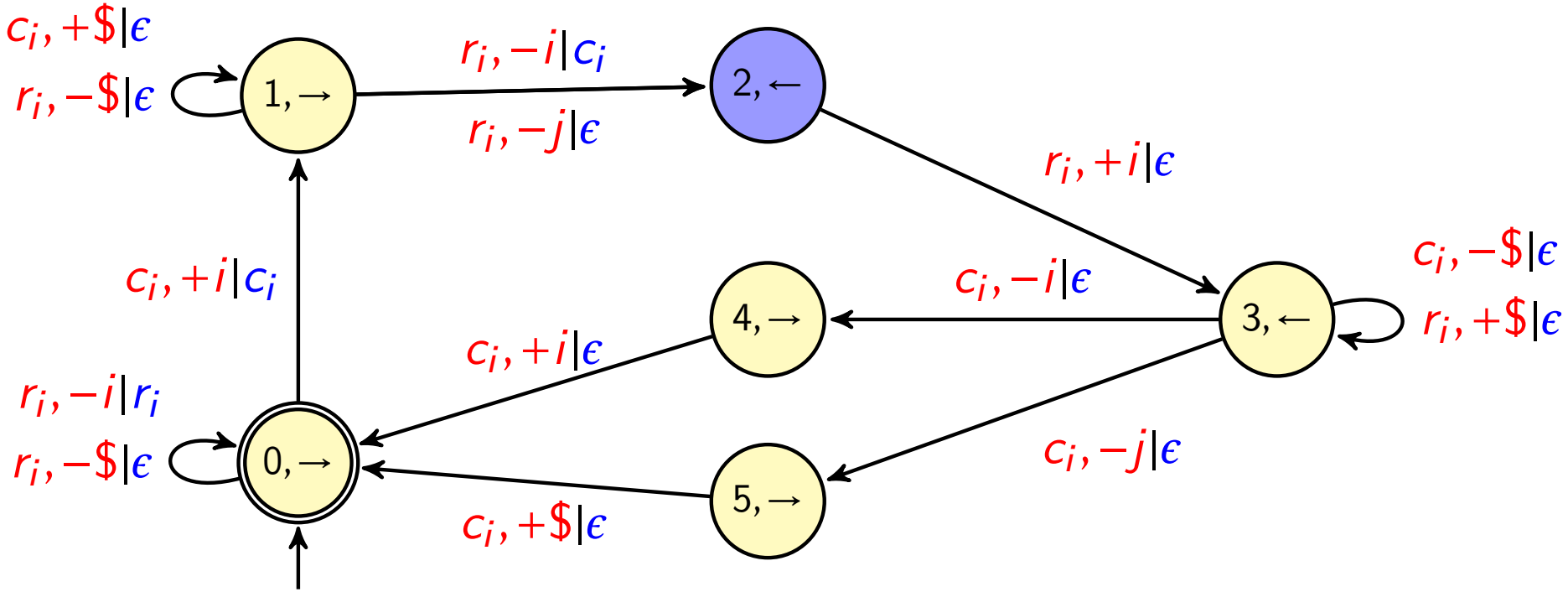
$(1, \rightarrow)$

Output Tape: c_1



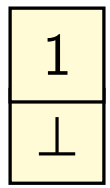
stack

Example: Deleting unequal matchings



Input Tape: $\vdash$ c_1 c_2 r_3 r_1 c_4 r_4 $\dashv$

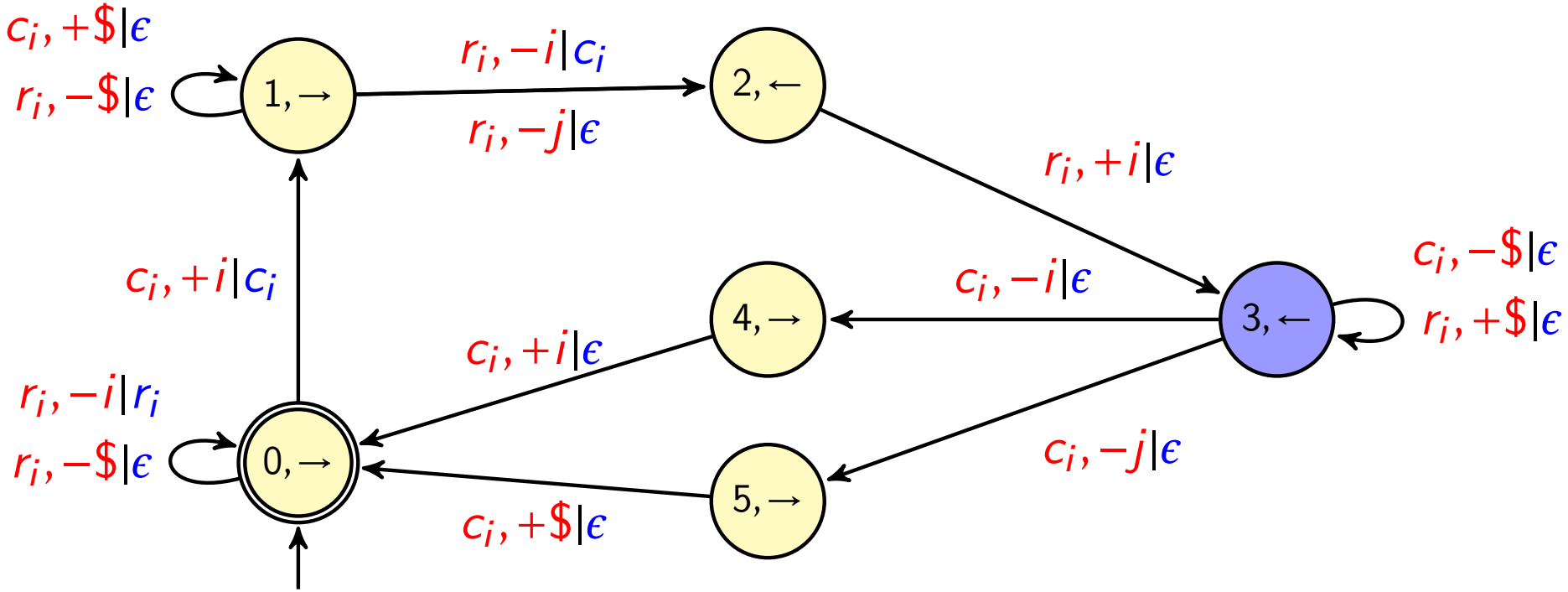
$(2, \leftarrow)$



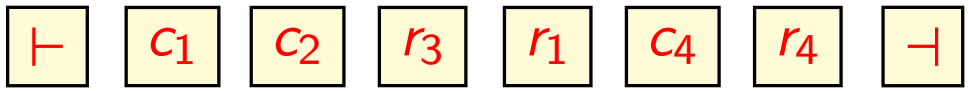
stack

Output Tape: c_1

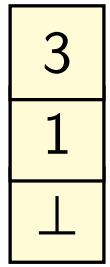
Example: Deleting unequal matchings



Input Tape:

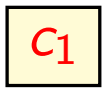


$(3, \leftarrow)$

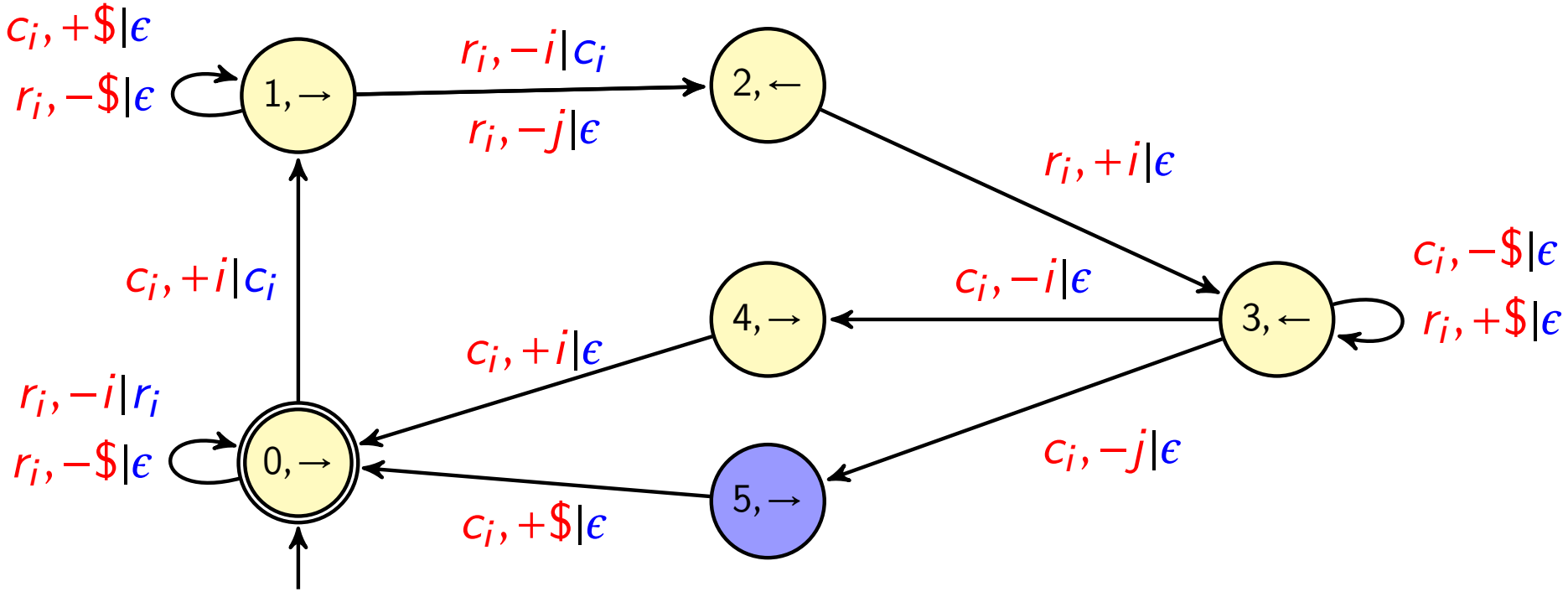


stack

Output Tape:



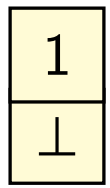
Example: Deleting unequal matchings



Input Tape: $\vdash$ c_1 c_2 r_3 r_1 c_4 r_4 $\dashv$

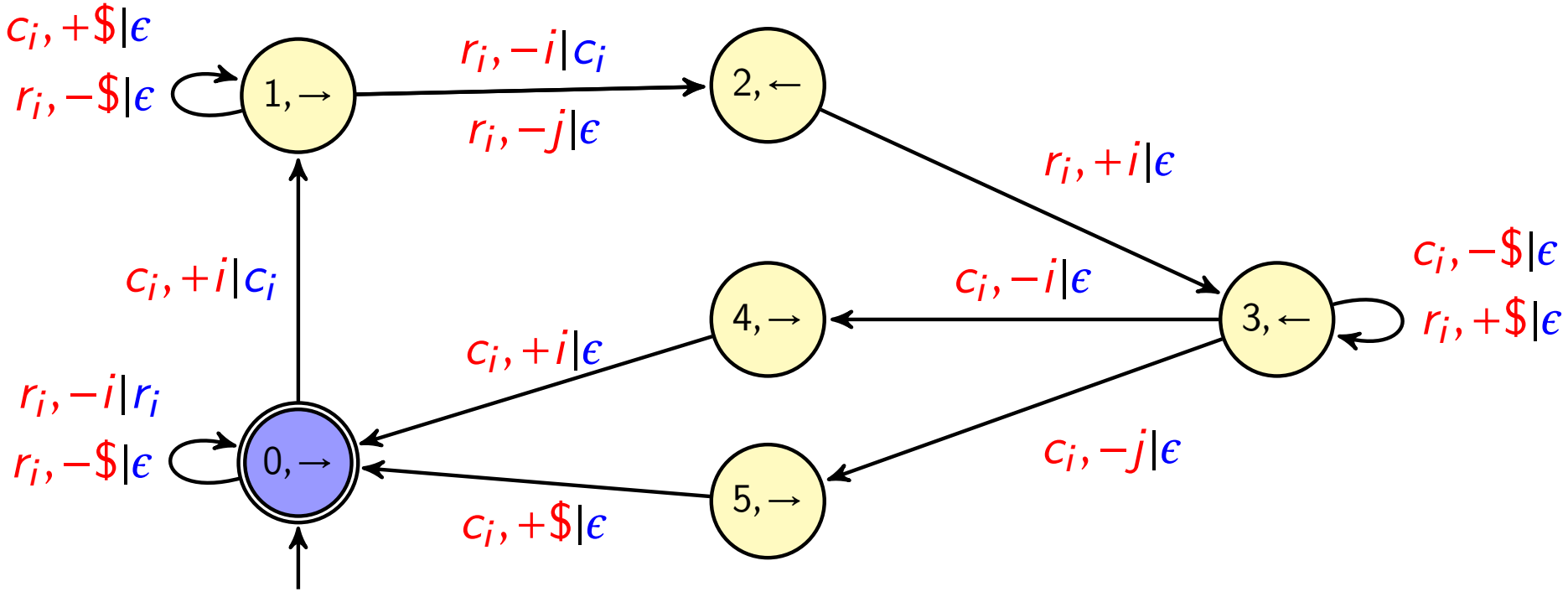
$(5, \rightarrow)$

Output Tape: c_1



stack

Example: Deleting unequal matchings



Input Tape: ⊢ c_1 c_2 r_3 r_1 c_4 r_4 ⊣

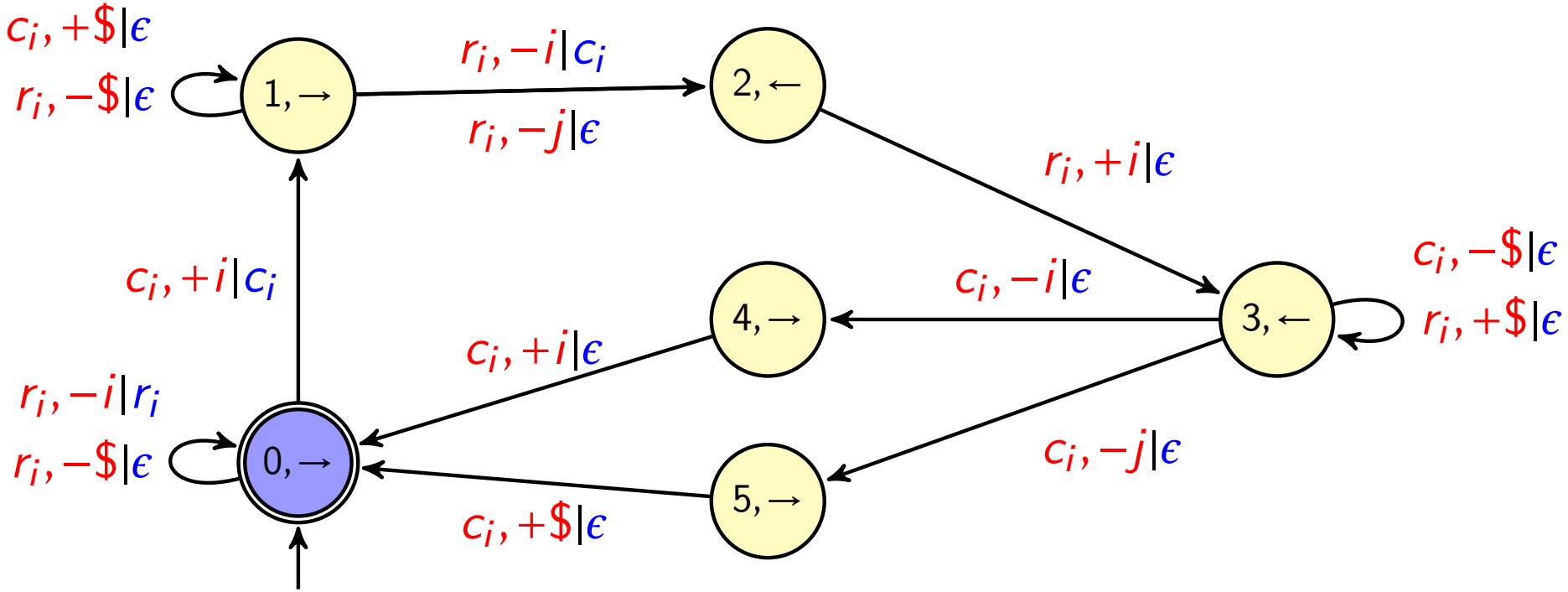
(0, →)

Output Tape: c_1

$
1
⊥

stack

Example: Deleting unequal matchings



Input Tape: $\vdash$ c_1 c_2 r_3 r_1 c_4 r_4 $\dashv$

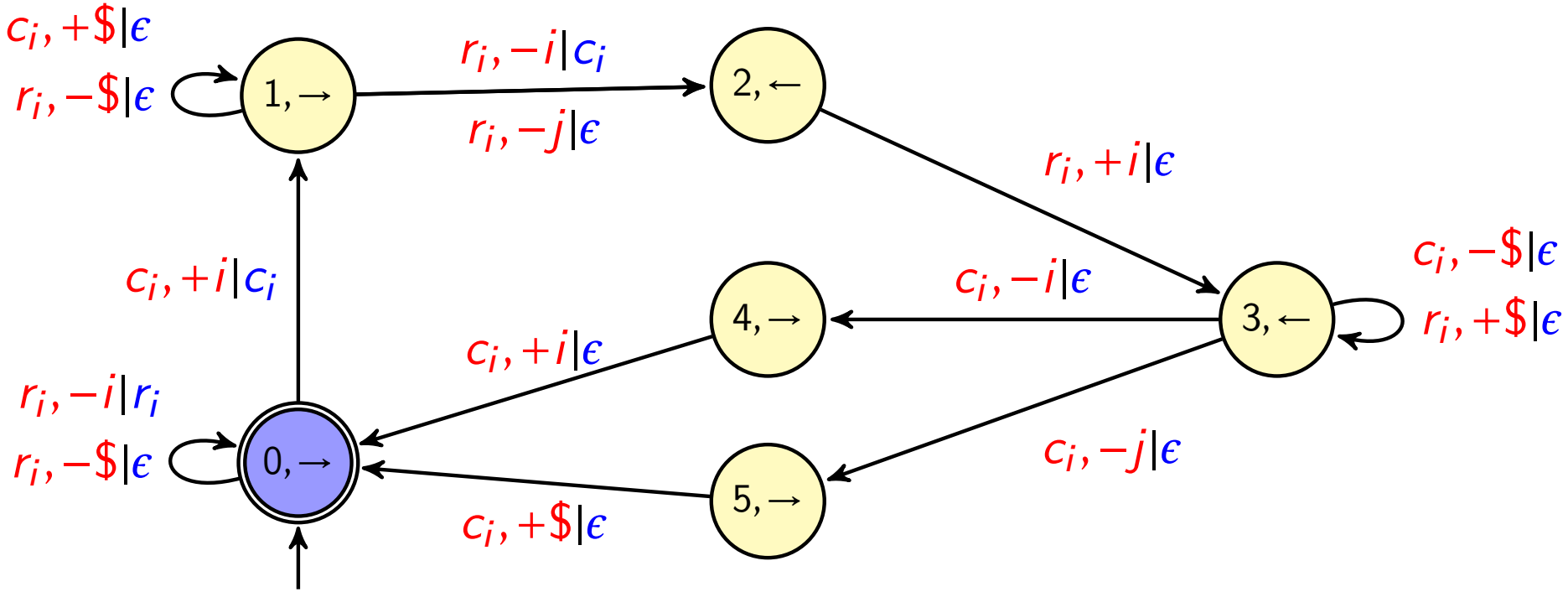
$(0, \rightarrow)$

1
$\perp$

stack

Output Tape: c_1

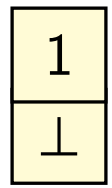
Example: Deleting unequal matchings



Input Tape: $\vdash$ c_1 c_2 r_3 r_1 c_4 r_4 $\dashv$

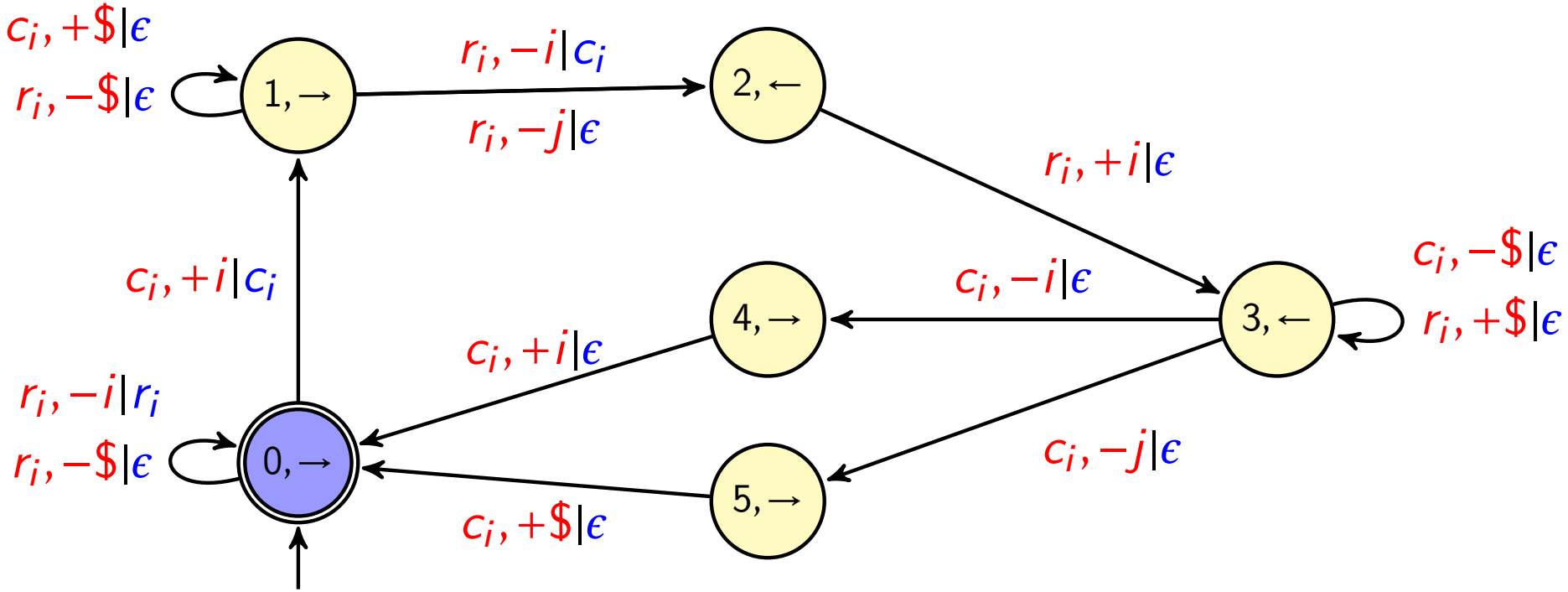
$(0, \rightarrow)$

Output Tape: c_1 r_1



stack

Example: Deleting unequal matchings



Input Tape: $\vdash$ c_1 c_2 r_3 r_1 c_4 r_4 $\dashv$

$(0, \rightarrow)$

$\perp$

stack

Output Tape: c_1 r_1 c_4 r_4

Expressiveness results

Theorem

$\mathrm{D2VPT}_{\mathrm{su}}$ are as expressive as MSO[nw2w].

Expressiveness results

Theorem

$\mathrm{D2VPT}_{su}$ are as expressive as MSO[nw2w].

Main proof Ingredient: Courcelle - Engelfriet 2012

$$\mathrm{DPTWT}^{la}_{lsi} = \mathrm{MSO}[\mathrm{BinaryTree2Word}]$$

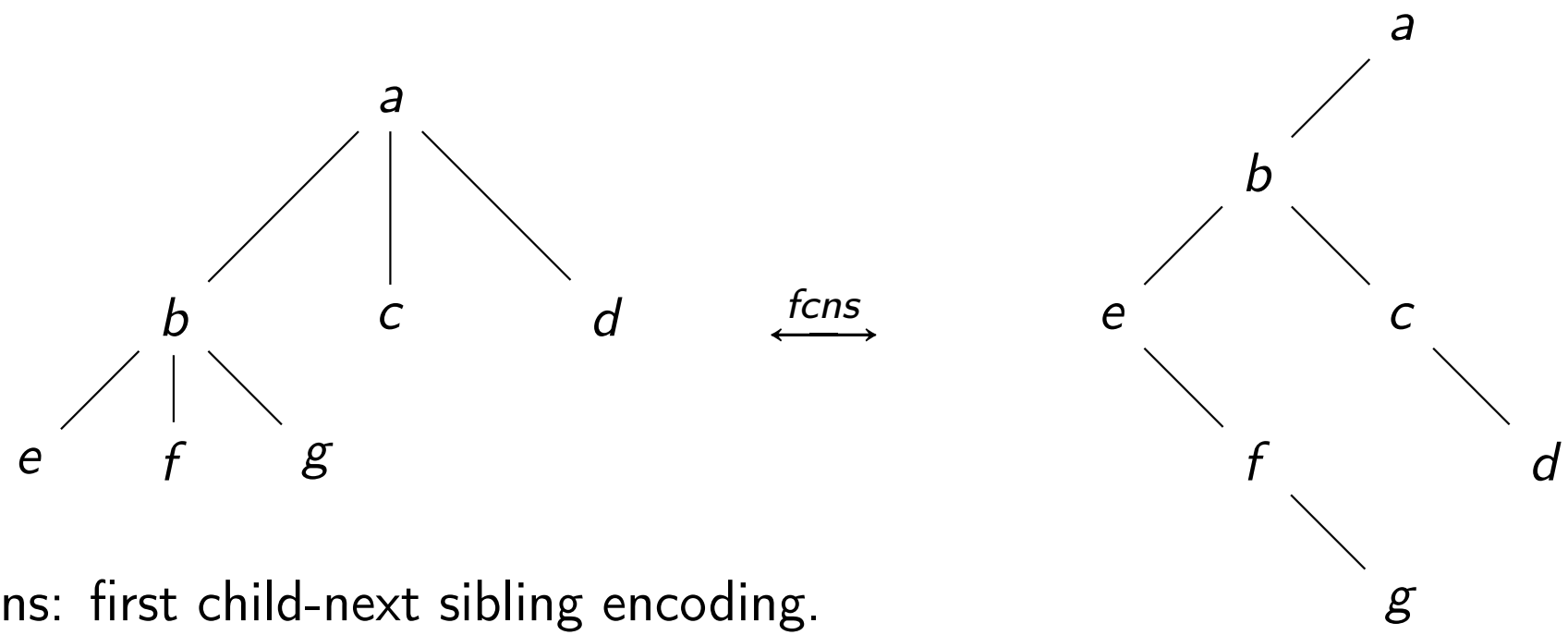
Expressiveness results

Theorem

$\mathrm{D2VPT}_{su}$ are as expressive as MSO[nw2w].

Main proof Ingredient: Courcelle - Engelfriet 2012

$$\mathrm{DPTWT}^{la}_{lsi} = \mathrm{MSO[BinaryTree2Word]}$$



fcns: first child-next sibling encoding.

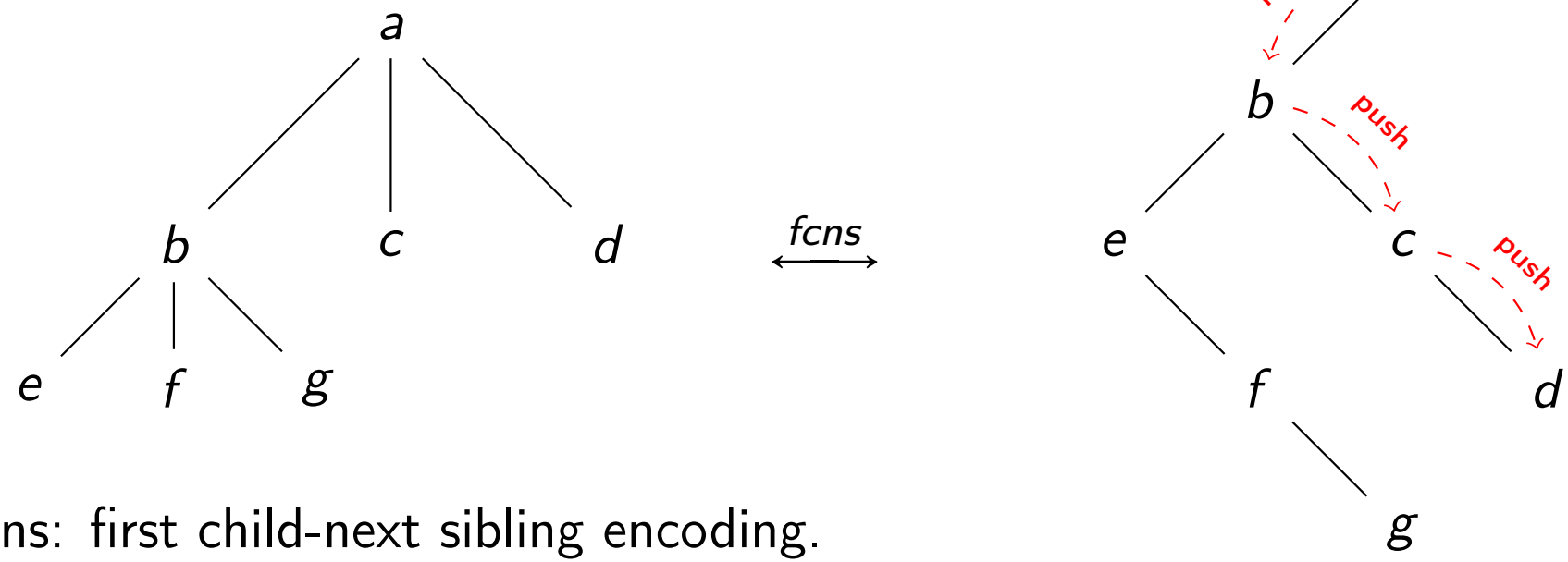
Expressiveness results

Theorem

$\mathrm{D2VPT}_{su}$ are as expressive as MSO[nw2w].

Main proof Ingredient: Courcelle - Engelfriet 2012

$$\mathrm{DPTWT}^{la}_{lsi} = \mathrm{MSO[BinaryTree2Word]}$$



fcns: first child-next sibling encoding.

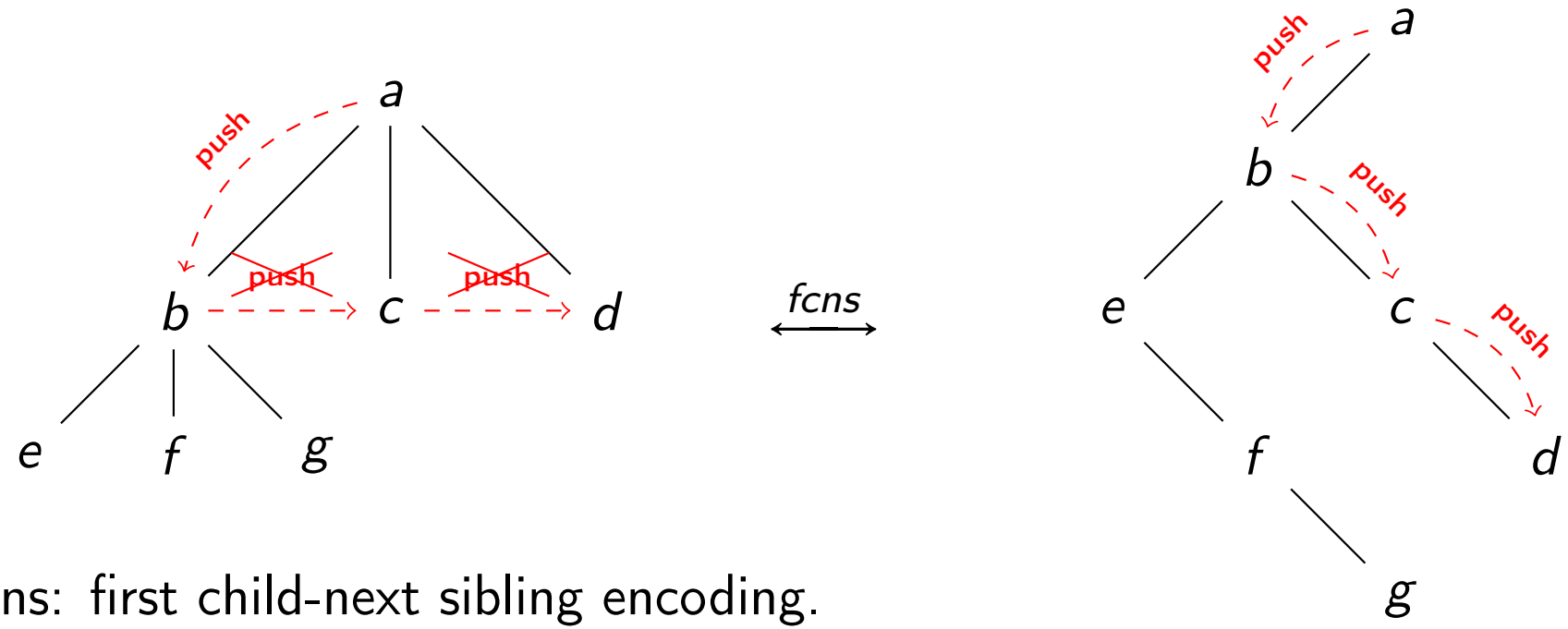
Expressiveness results

Theorem

$\mathrm{D2VPT}_{su}$ are as expressive as MSO[nw2w].

Main proof Ingredient: Courcelle - Engelfriet 2012

$$\mathrm{DPTWT}^{la}_{lsi} = \mathrm{MSO[BinaryTree2Word]}$$



fcns: first child-next sibling encoding.

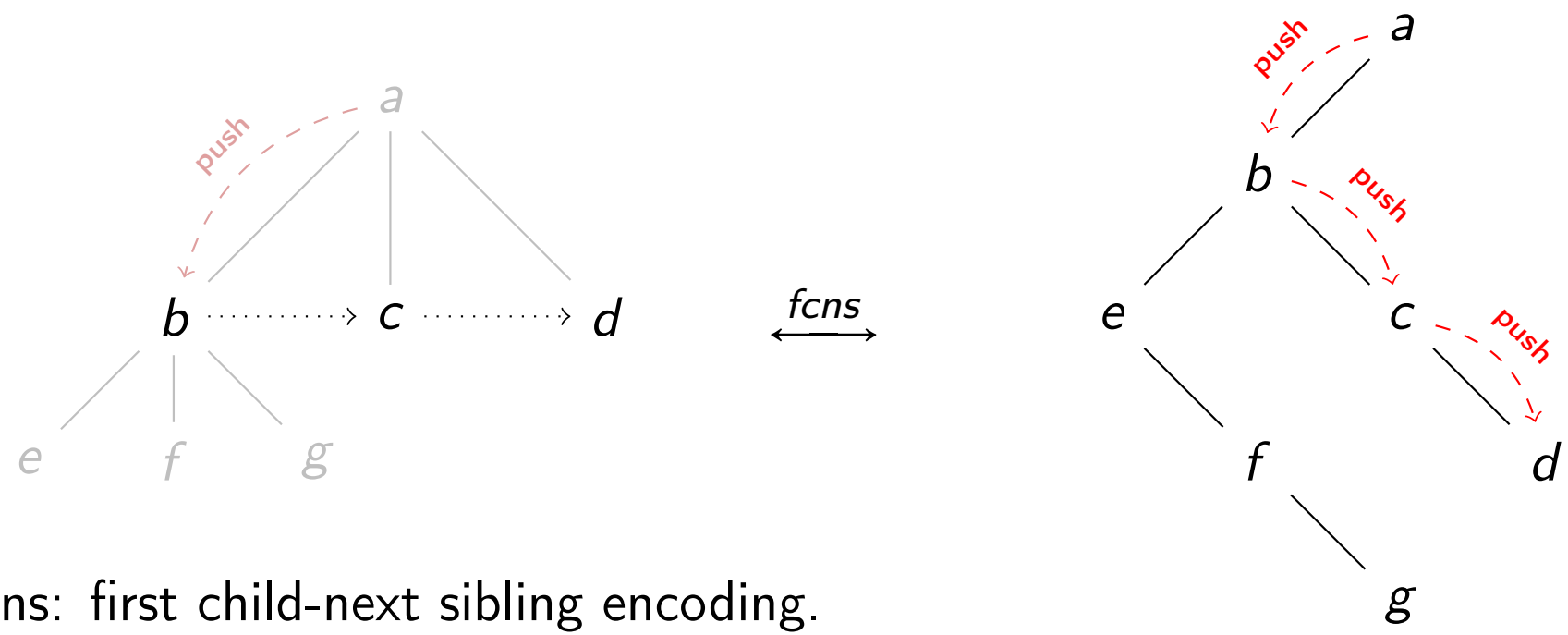
Expressiveness results

Theorem

$\mathrm{D2VPT}_{su}$ are as expressive as MSO[nw2w].

Main proof Ingredient: Courcelle - Engelfriet 2012

$$\mathrm{DPTWT}^{la}_{lsi} = \mathrm{MSO[BinaryTree2Word]}$$



fcns: first child-next sibling encoding.

Conclusion

The class of D2VPT

- Natural model with good evaluation complexity,
- Decidable equivalence,
- Captures MSO[nw2w],
- Closed under look-around,
- Models unranked tree to word transductions.

Open questions

- Nested Word to Nested Word: Decide if the range of a D2VPT is well-nested?
- Following up: Full closure under composition?
- Streaming: Decide if a D2VPT is VPT definable?