Multi-Stack Automata

Wojciech Czerwiński

Multi-Stack Automata

(Partially-Commutive Context-Free Graphs)

Wojciech Czerwiński

Plan

Plan

- what are Multi-Stack Automata

Plan

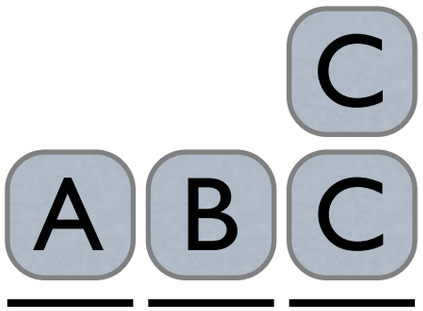
- what are Multi-Stack Automata
- why I investigate them

Plan

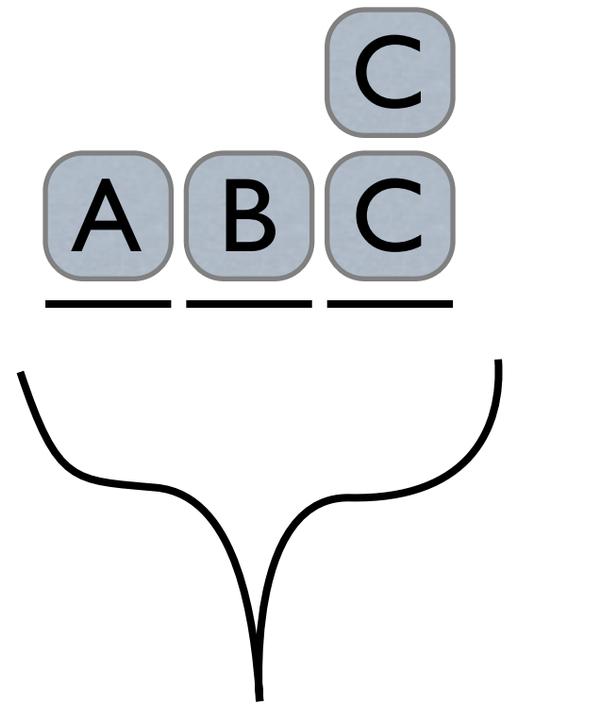
- what are Multi-Stack Automata
- why I investigate them
- what I have solved

Multi-Stack Automaton

Multi-Stack Automaton

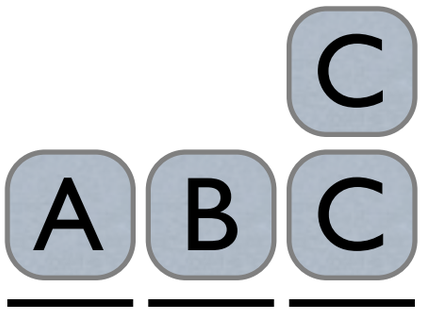


Multi-Stack Automaton

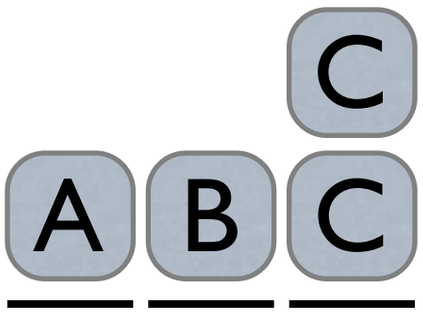


Configuration
(3 stacks)

Multi-Stack Automaton

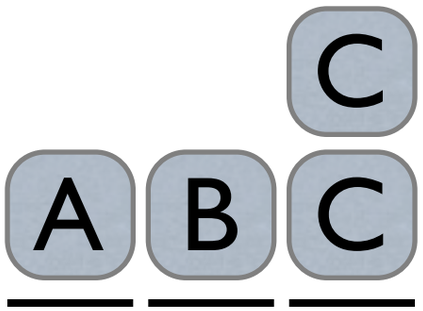


Multi-Stack Automaton



$C \longrightarrow (A, BD, C)$

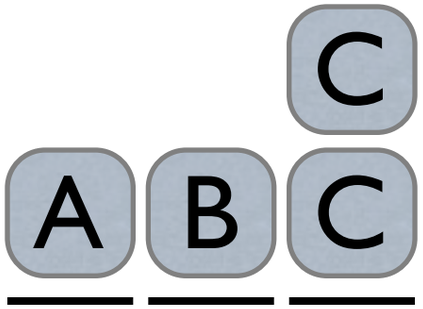
Multi-Stack Automaton



$C \longrightarrow (A, BD, C)$

$A \longrightarrow (\_, B, \_)$

Multi-Stack Automaton

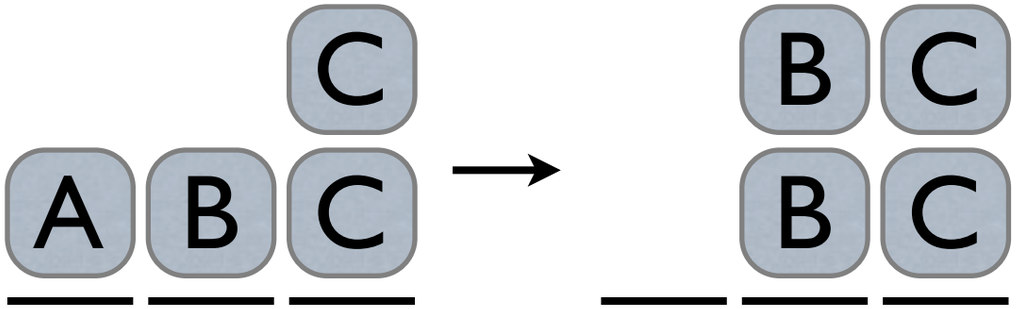


$C \longrightarrow (A, BD, C)$

$A \longrightarrow (\_, B, \_)$

$C \longrightarrow (\_, \_, \_)$

Multi-Stack Automaton

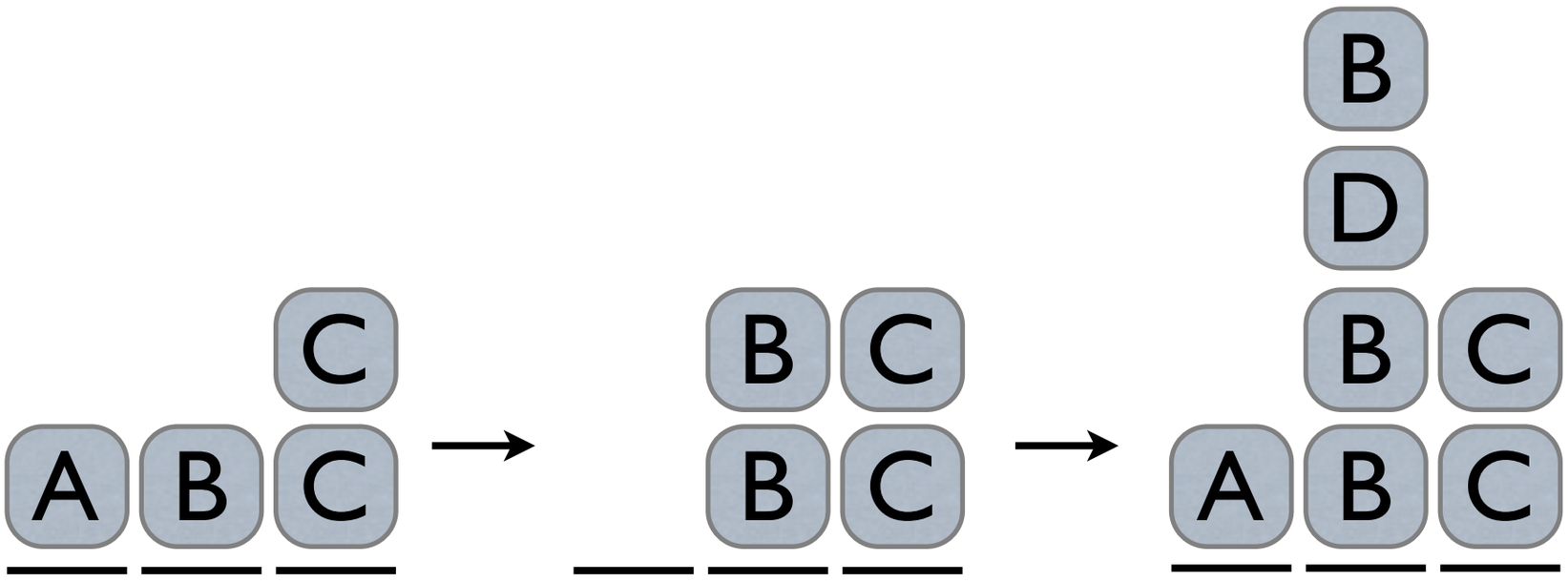


C ⟶ (A, BD, C)

A ⟶ (_, B, _)

C ⟶ (_, _, _)

Multi-Stack Automaton

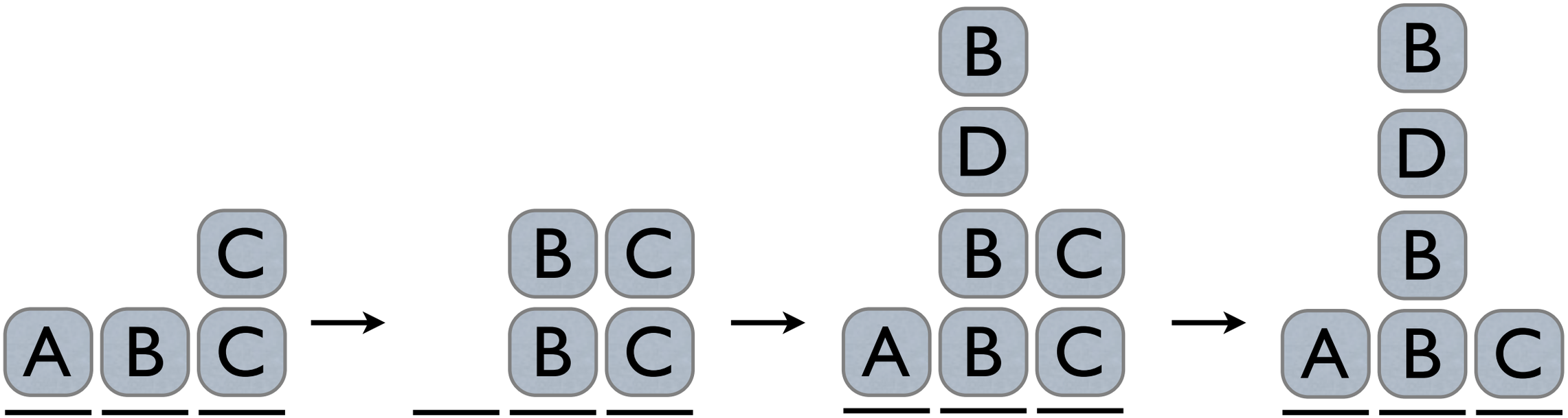


C ⟶ (A, BD, C)

A ⟶ (_, B, _)

C ⟶ (_, _, _)

Multi-Stack Automaton

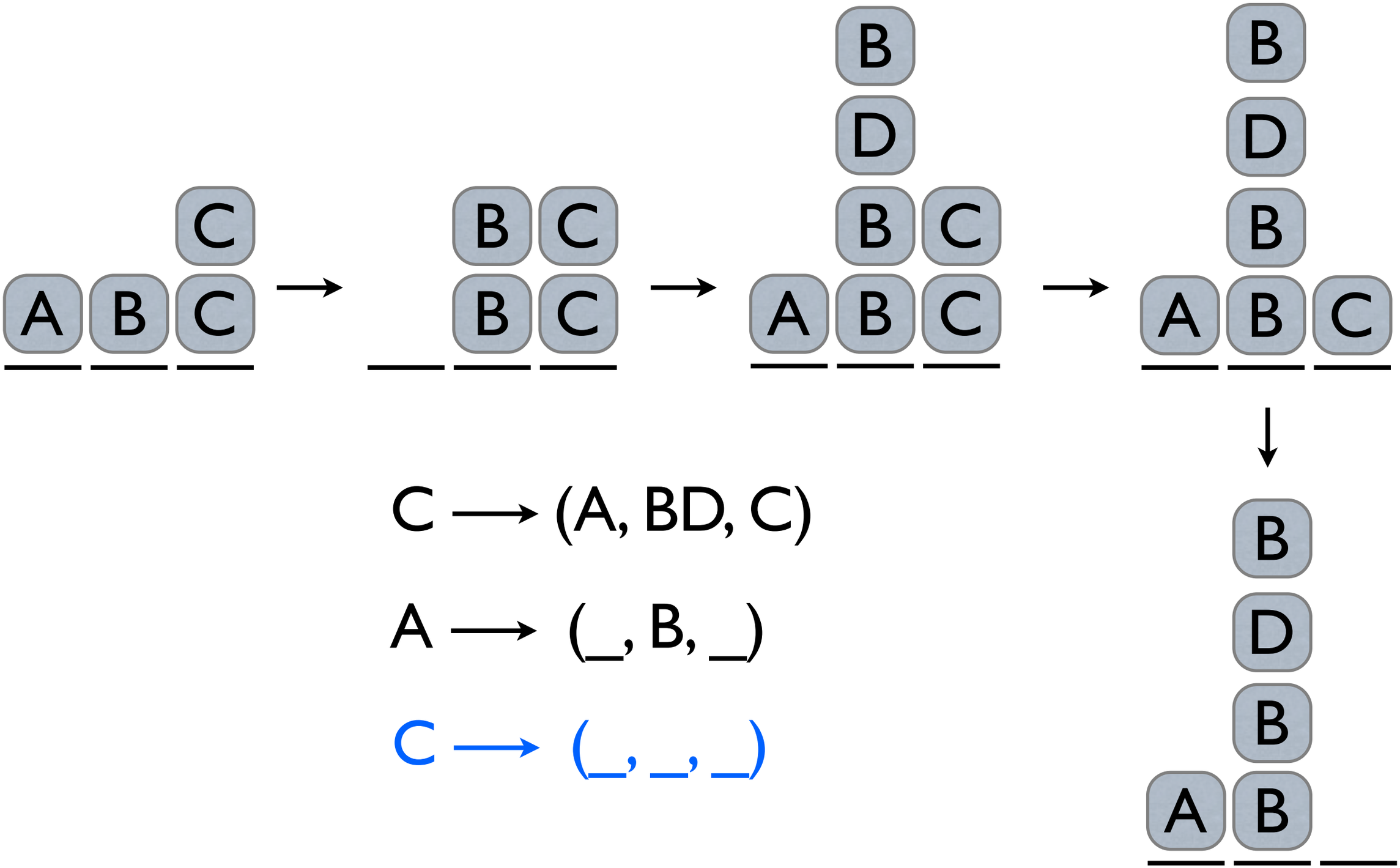


C ⟶ (A, BD, C)

A ⟶ (_, B, _)

C ⟶ (_, _, _)

Multi-Stack Automaton



C ⟶ (A, BD, C)

A ⟶ (_, B, _)

C ⟶ (_, _, _)

Motivation

Motivation

- errors in programs

Motivation

- errors in programs
- automatic detection of errors

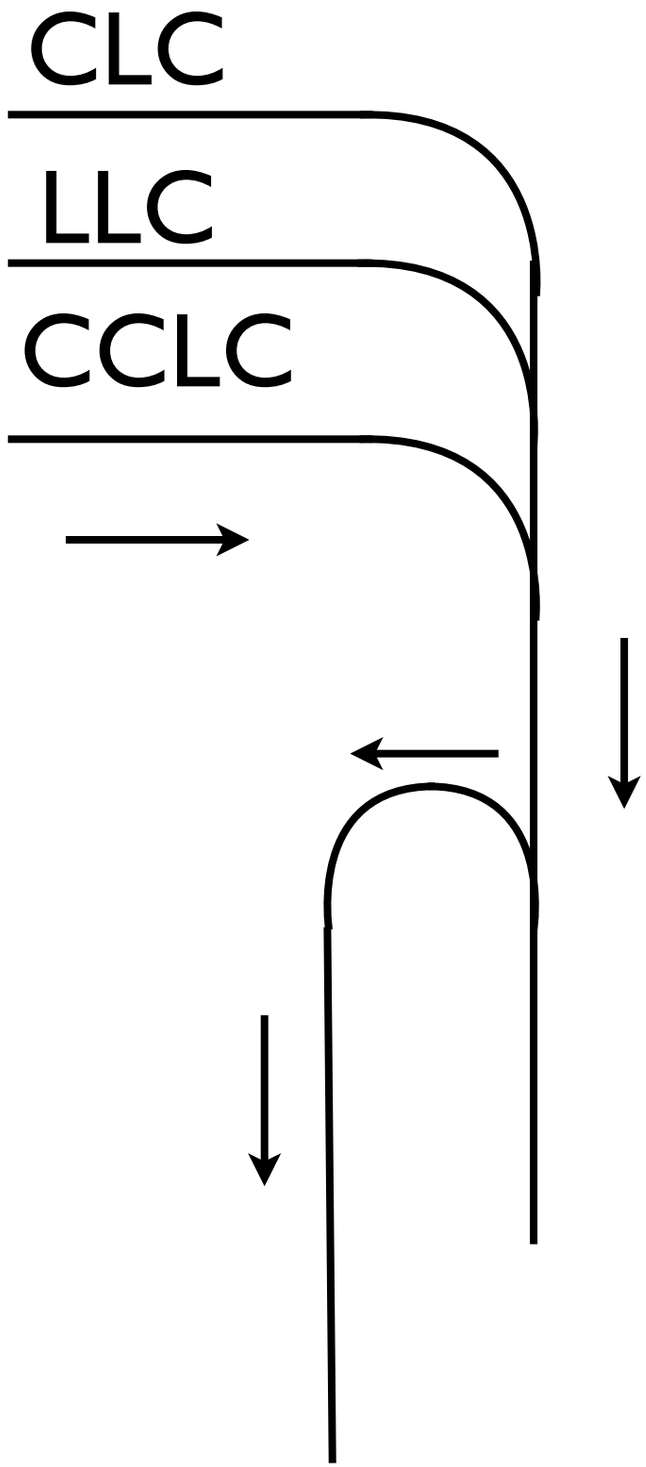
Motivation

- errors in programs
- automatic detection of errors
- model the program

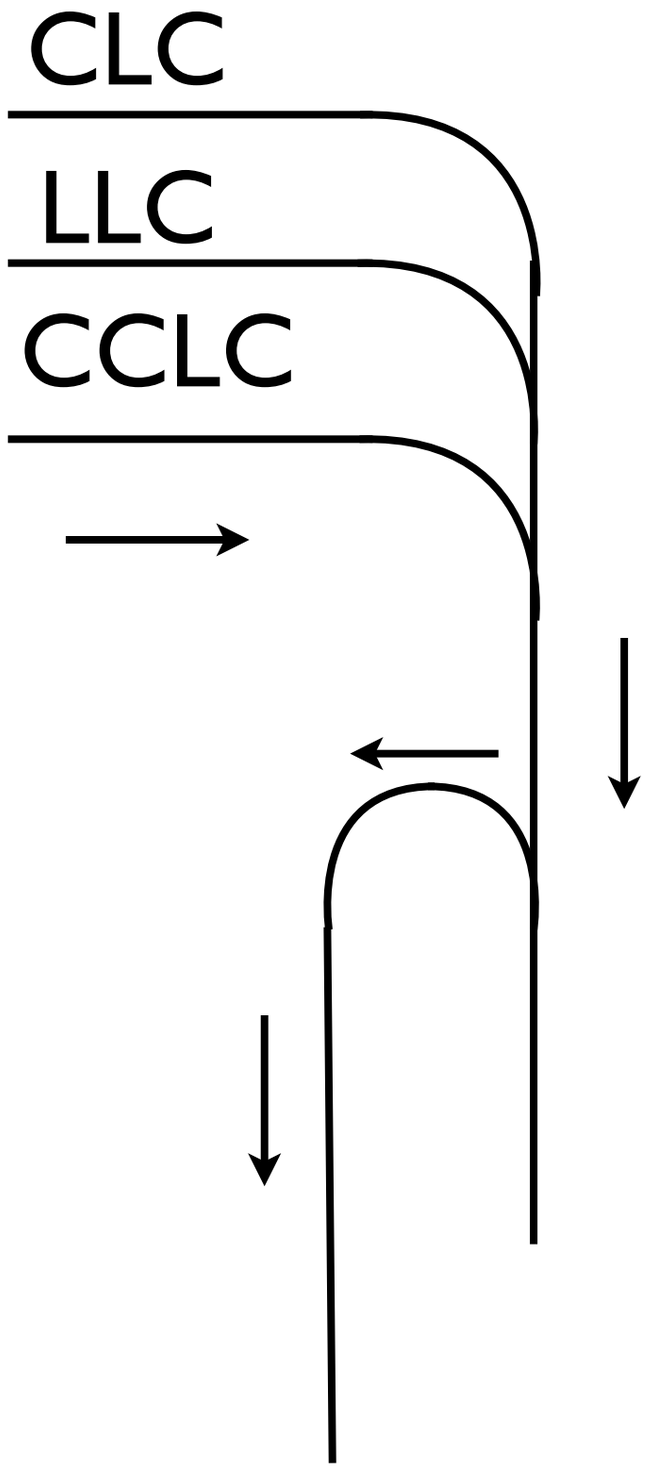
Motivation

- errors in programs
- automatic detection of errors
- model the program
- Multi-Stack Automata as a model

Real life example

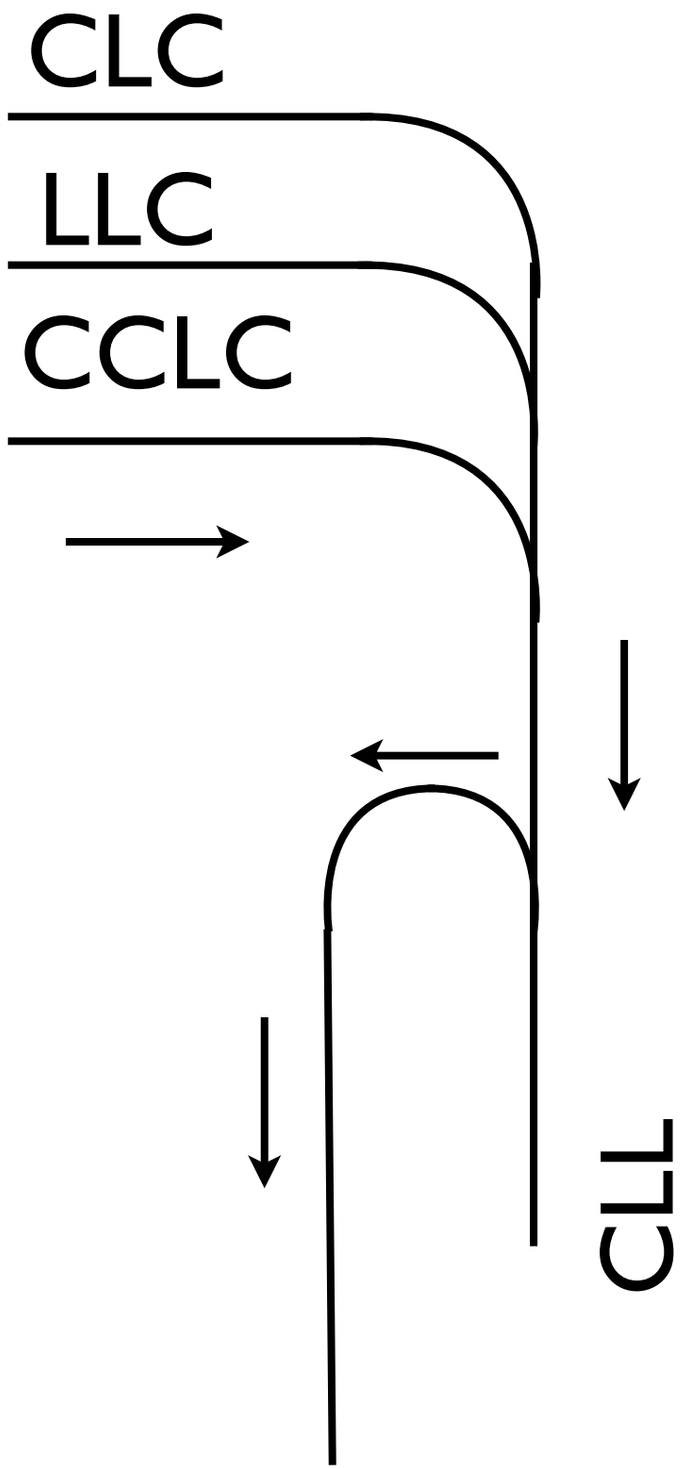


Real life example



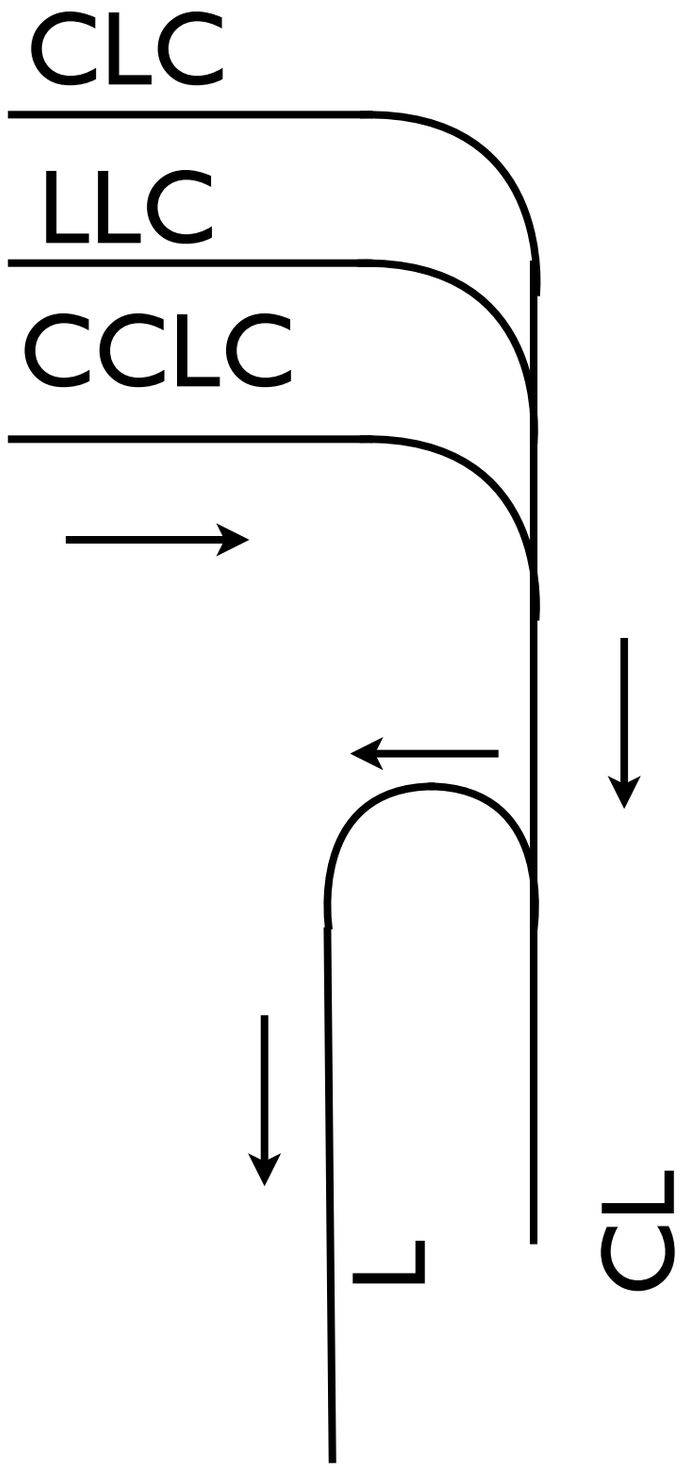
Can we produce trains with 2 cars and 1 locomotive?

Real life example



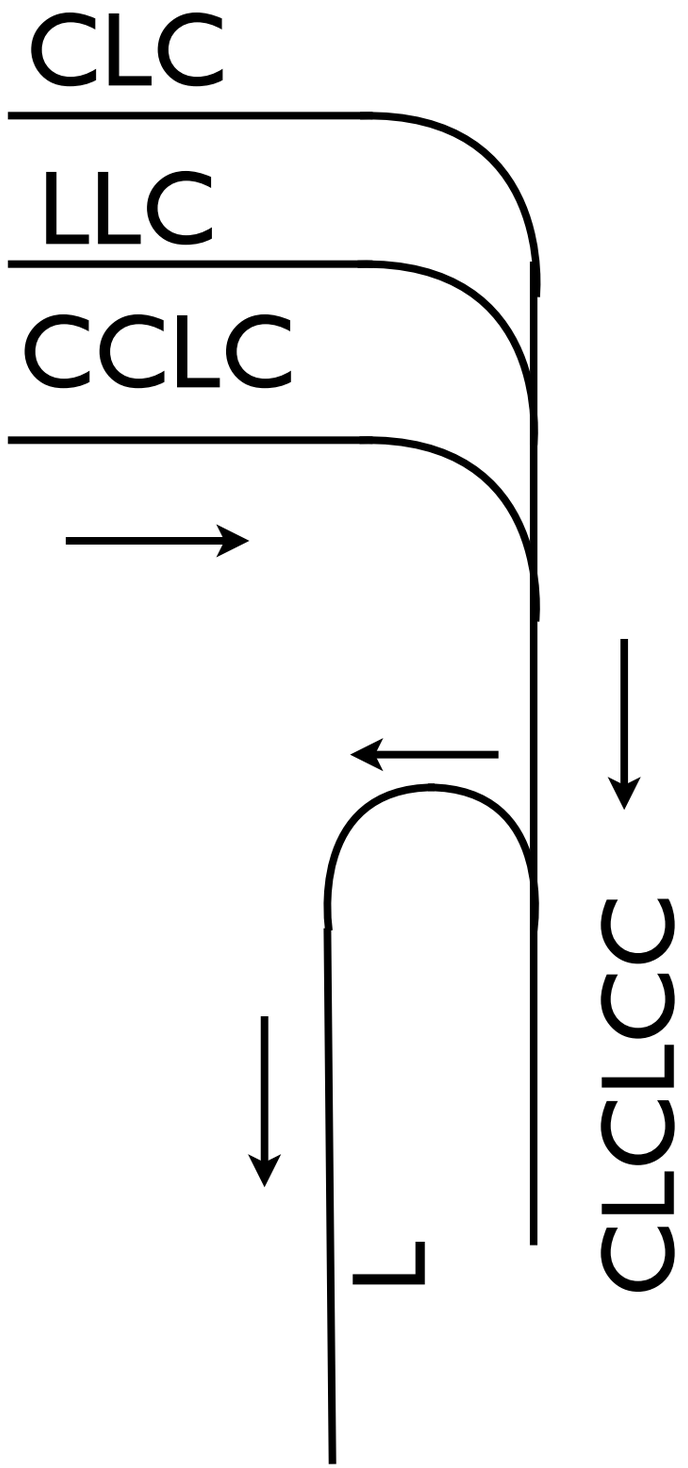
Can we produce trains with 2 cars and 1 locomotive?

Real life example



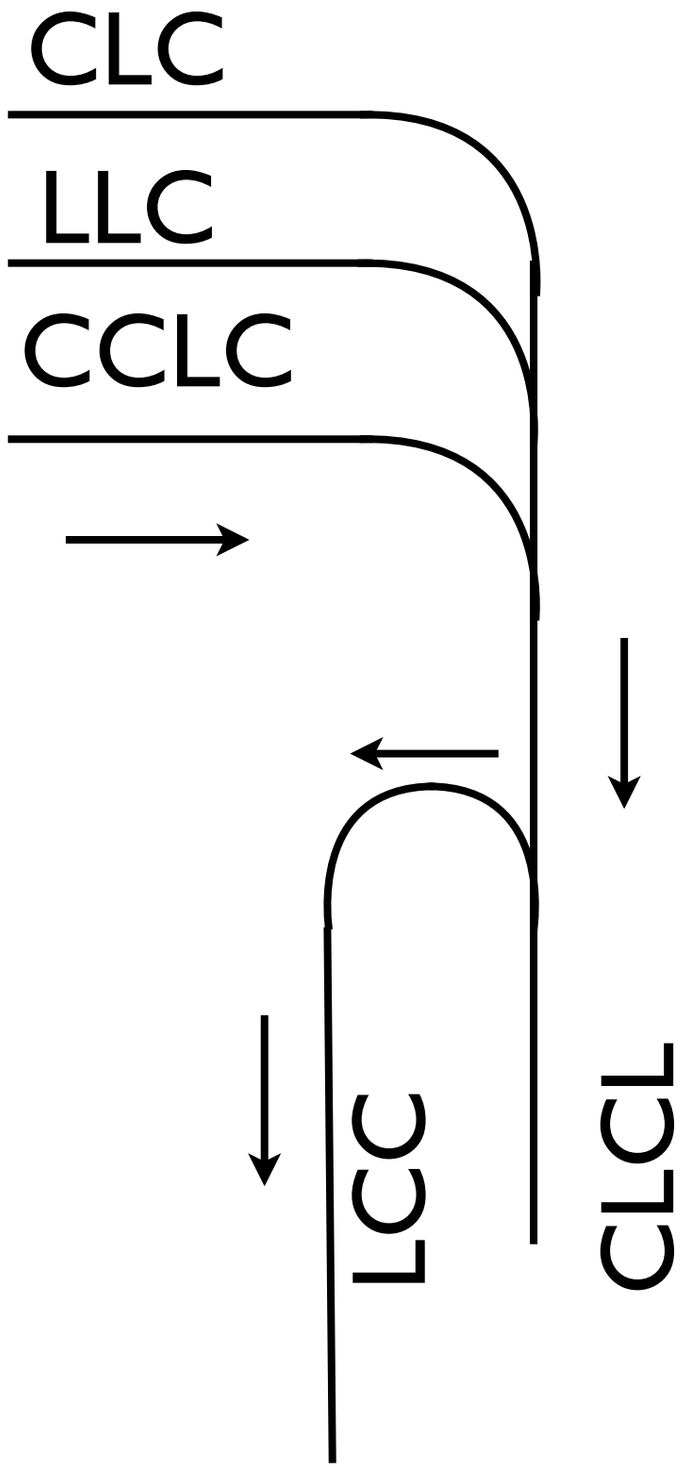
Can we produce trains with 2 cars and 1 locomotive?

Real life example



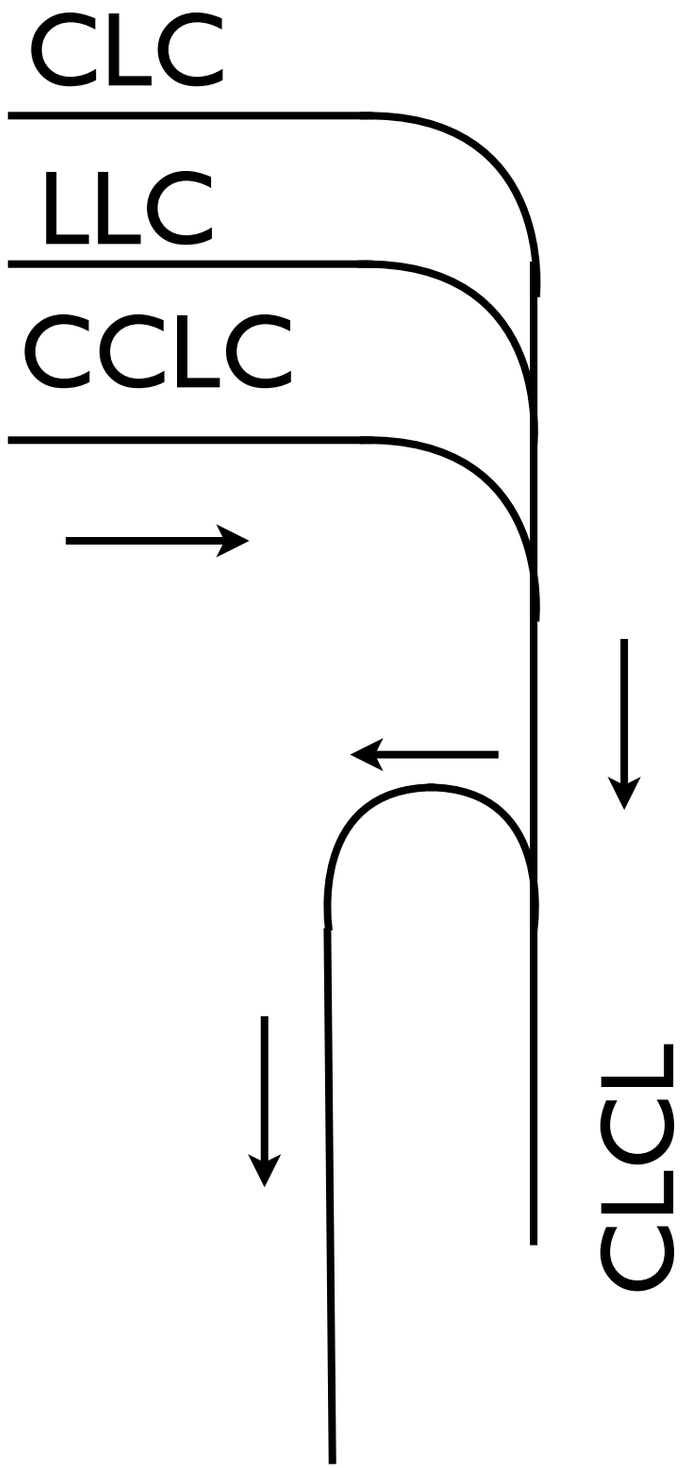
Can we produce trains with 2 cars and 1 locomotive?

Real life example



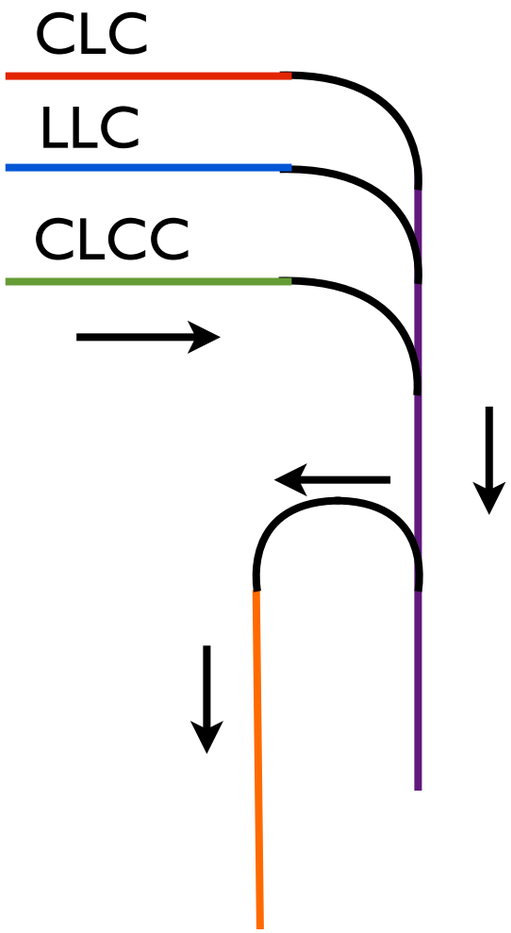
Can we produce trains with 2 cars and 1 locomotive?

Real life example



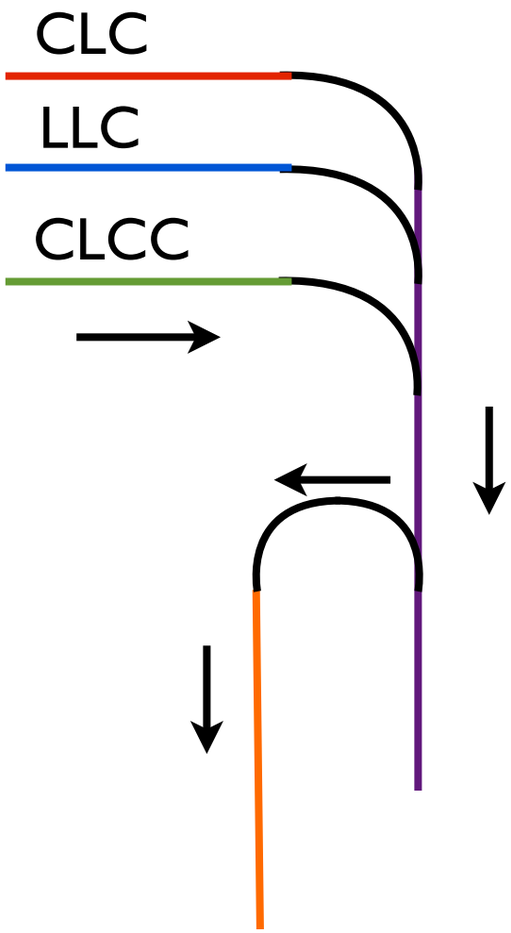
Can we produce trains with 2 cars and 1 locomotive?

Modelling

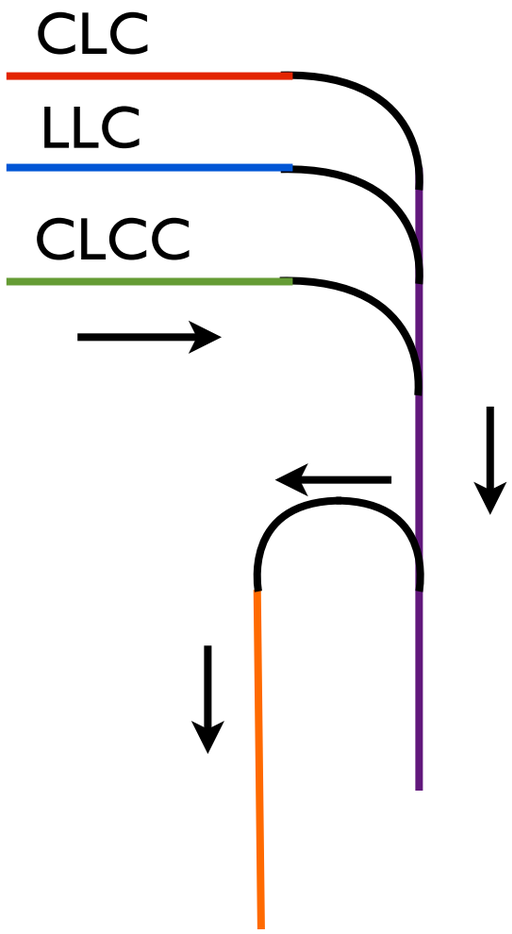


Modelling

$$T \longrightarrow (T, \_, \_, CLC, \_)$$



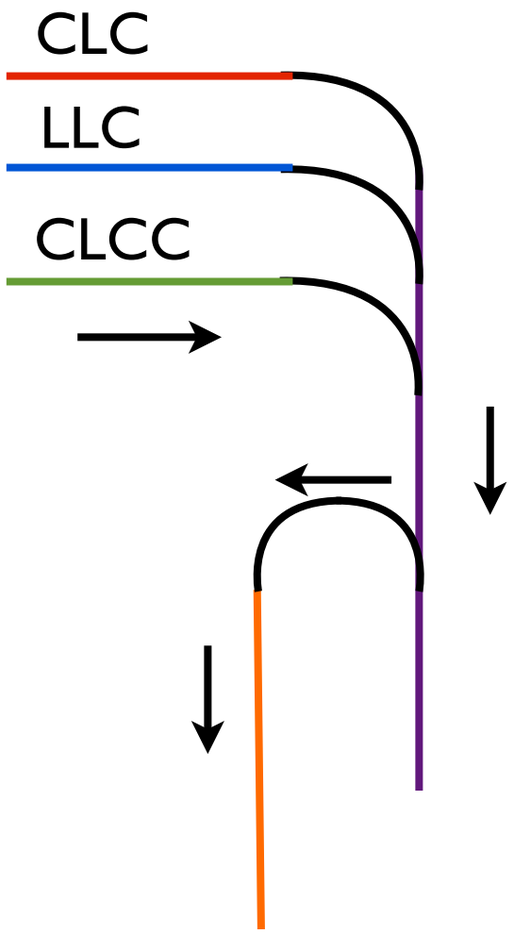
Modelling



T ⟶ (T, _, _, CLC, _)

T ⟶ (_, T, _, LLC, _)

Modelling

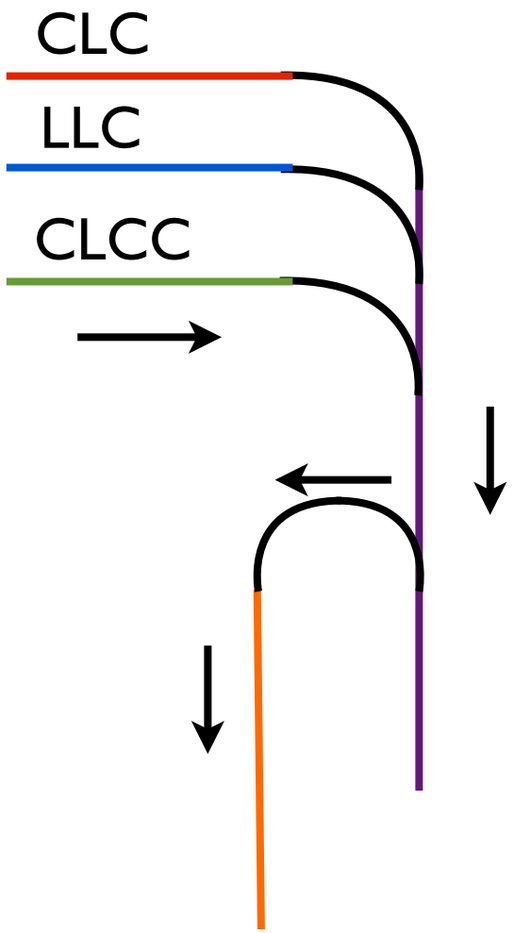


T ⟶ (T, _, _, CLC, _)

T ⟶ (_, T, _, LLC, _)

T ⟶ (_, _, T, CLCC, _)

Modelling



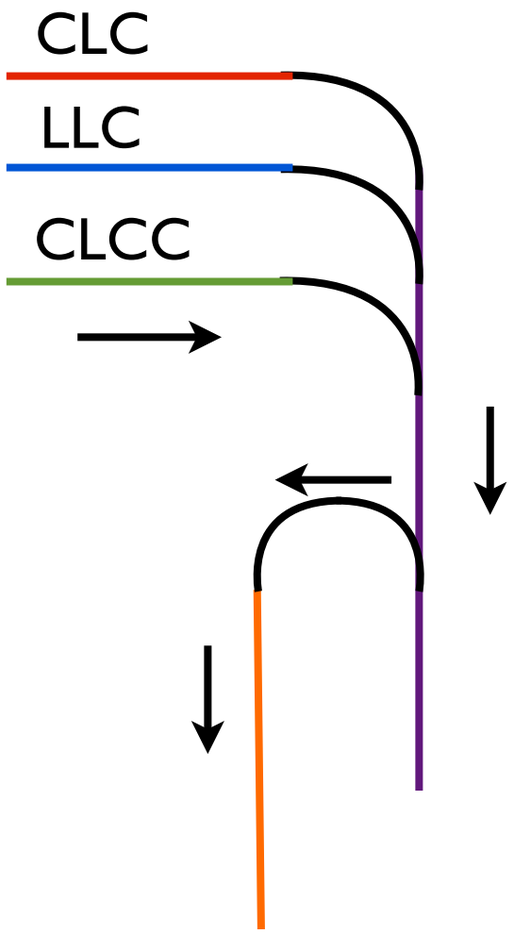
$T \longrightarrow (T, \_, \_, CLC, \_)$

$T \longrightarrow (\_, T, \_, LLC, \_)$

$T \longrightarrow (\_, \_, T, CLCC, \_)$

$C \longrightarrow (\_, \_, \_, \_, C)$

Modelling



$T \longrightarrow (T, \_, \_, CLC, \_)$

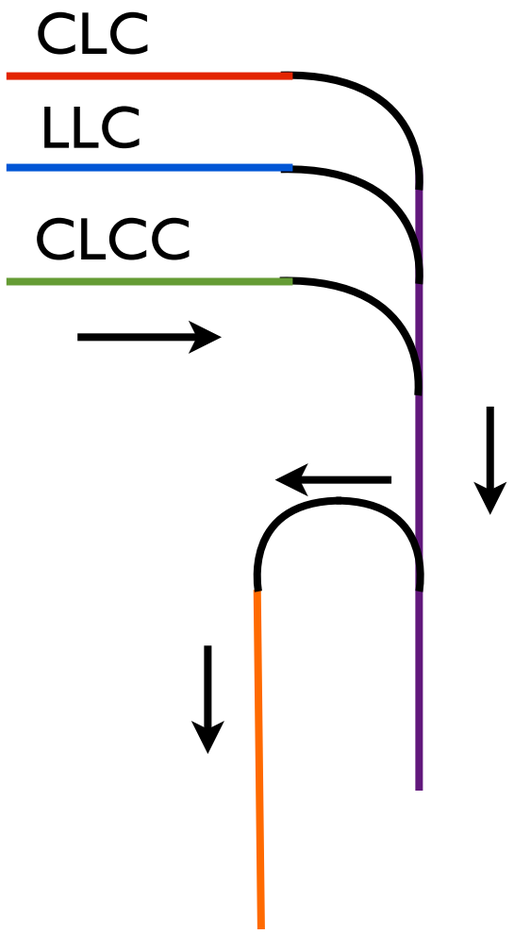
$T \longrightarrow (\_, T, \_, LLC, \_)$

$T \longrightarrow (\_, \_, T, CLCC, \_)$

$C \longrightarrow (\_, \_, \_, \_, C)$

$L \longrightarrow (\_, \_, \_, \_, L)$

Modelling



(T, T, T, _, _)

T ⟶ (T, _, _, CLC, _)

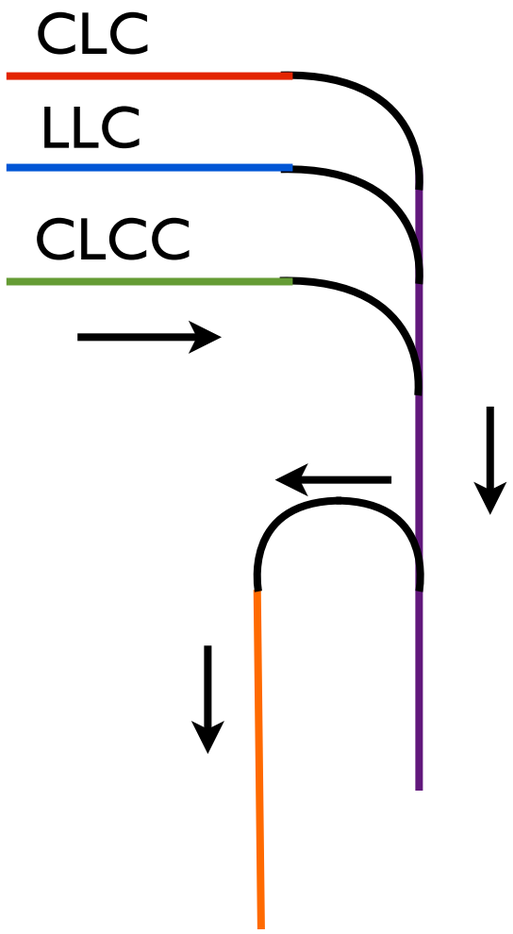
T ⟶ (_, T, _, LLC, _)

T ⟶ (_, _, T, CLCC, _)

C ⟶ (_, _, _, _, C)

L ⟶ (_, _, _, _, L)

Modelling



T ⟶ (T, _, _, CLC, _)

T ⟶ (_, T, _, LLC, _)

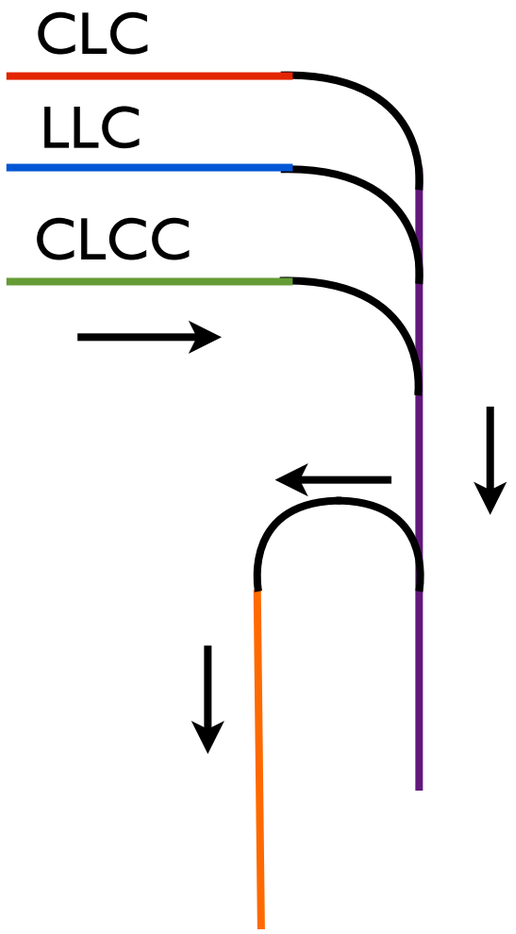
T ⟶ (_, _, T, CLCC, _)

C ⟶ (_, _, _, _, C)

L ⟶ (_, _, _, _, L)

(T, T, T, _, _) ⟶ ⟶ ? ... ⟶

Modelling



$T \longrightarrow (T, \_, \_, CLC, \_)$

$T \longrightarrow (\_, T, \_, LLC, \_)$

$T \longrightarrow (\_, \_, T, CLCC, \_)$

$C \longrightarrow (\_, \_, \_, \_, C)$

$L \longrightarrow (\_, \_, \_, \_, L)$

$(T, T, T, \_, \_) \longrightarrow \longrightarrow \overset{?}{\dots} \longrightarrow (T, T, T, \_, (CCL)^{+})$

Problems

Problems

Expressivity

Problems

Expressivity

Reachability problem

Problems

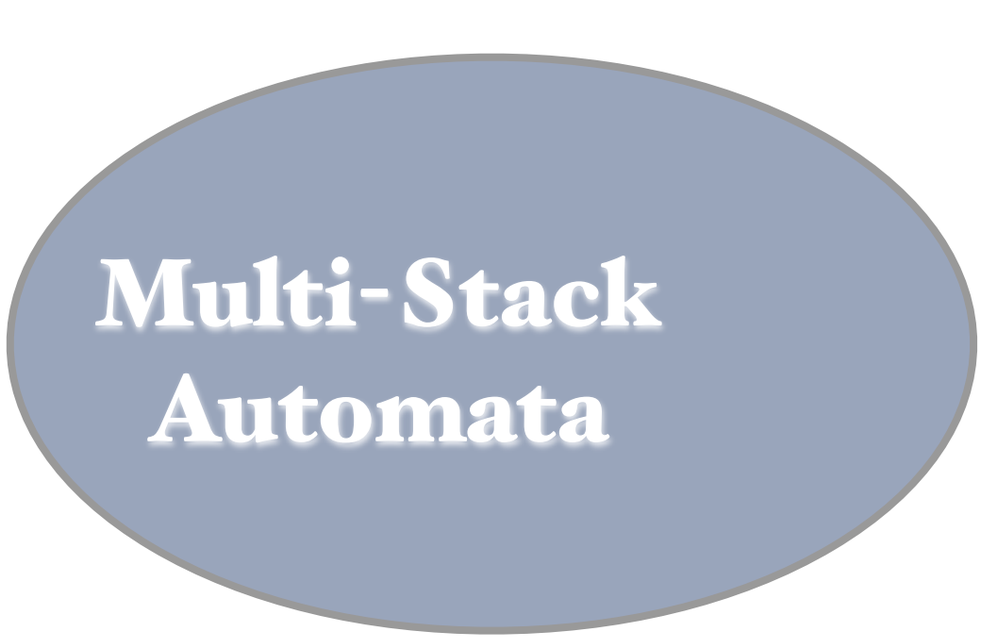
Expressivity

Reachability problem

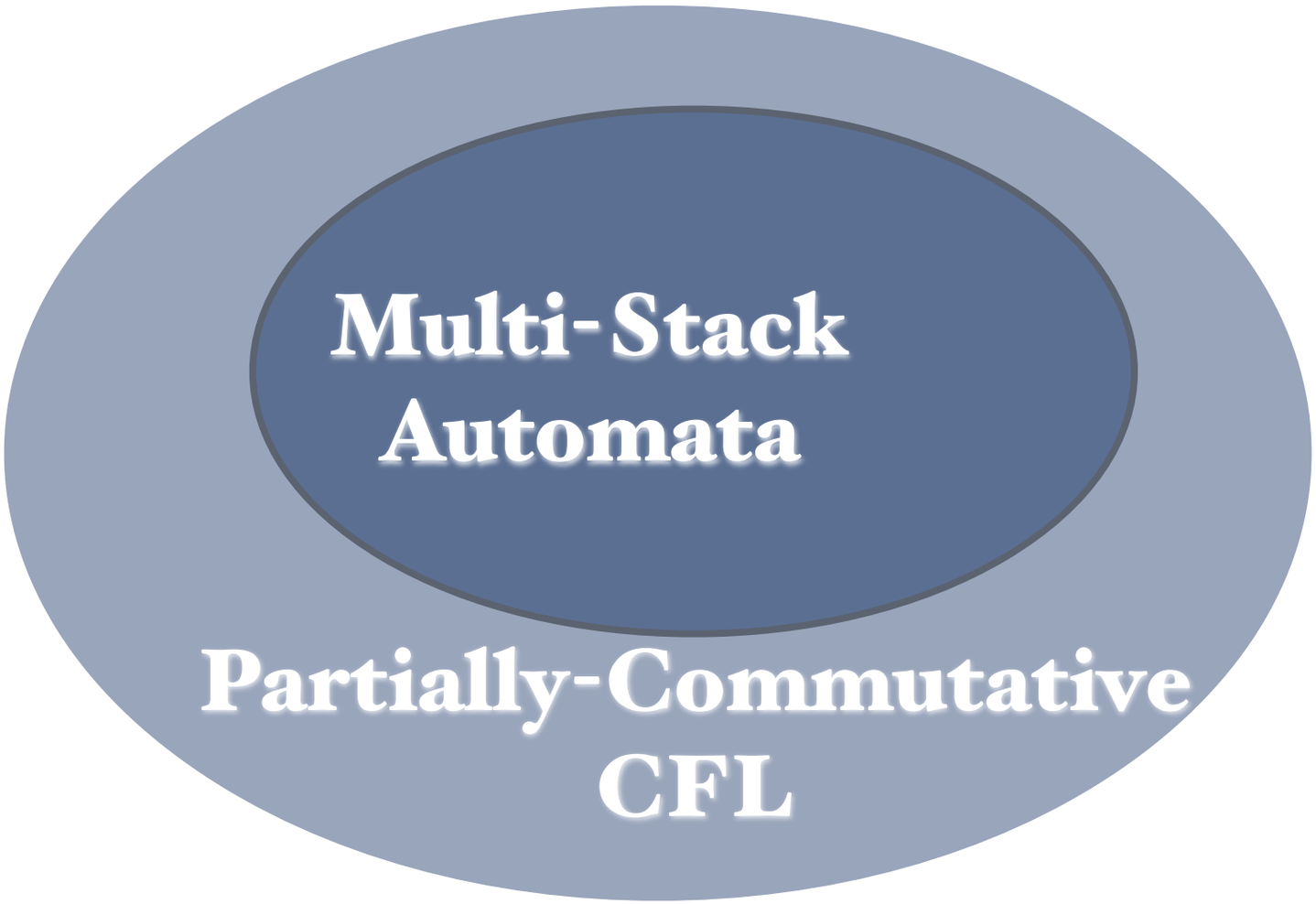
Equivalence problem

Main result (expressivity)

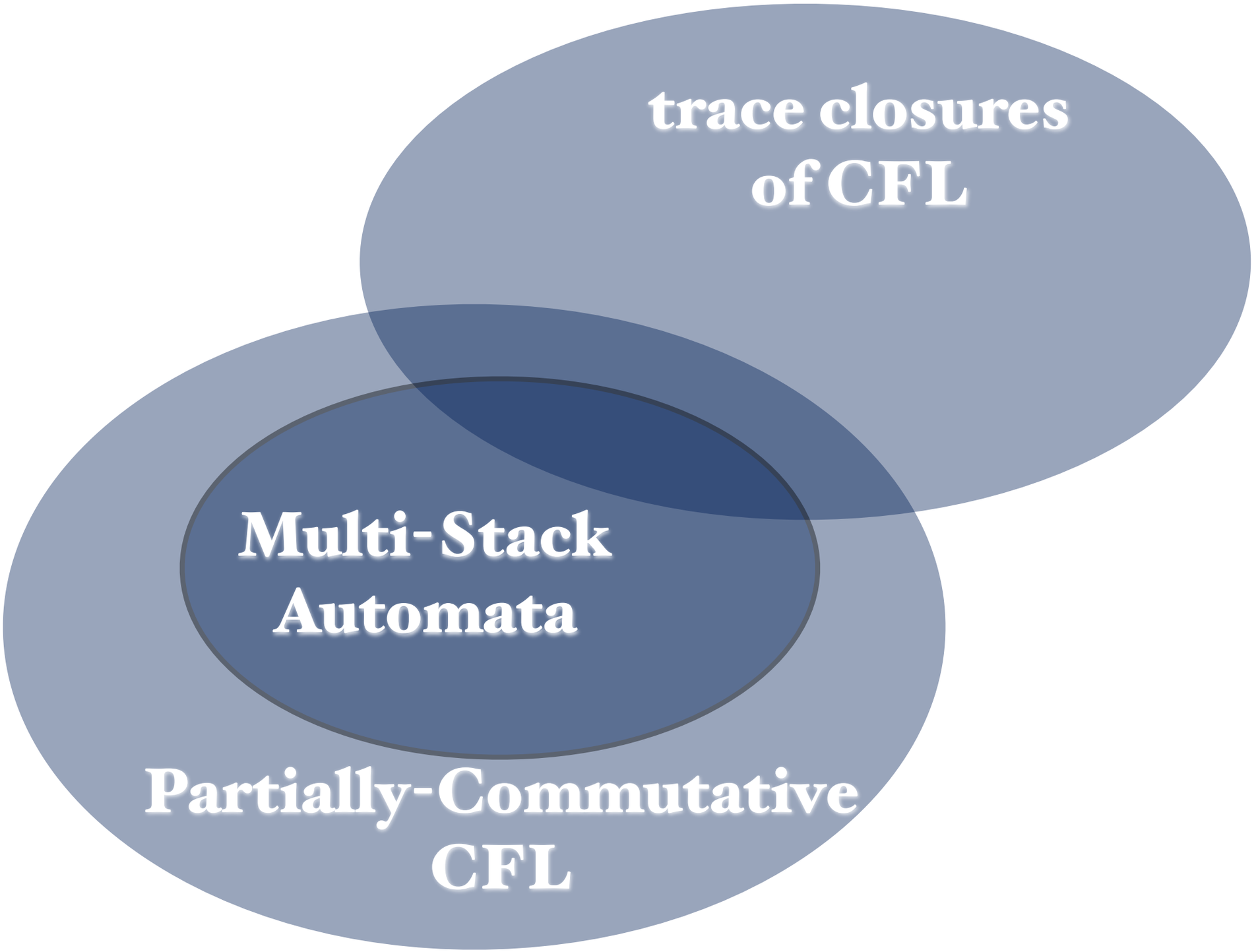
Main result (expressivity)



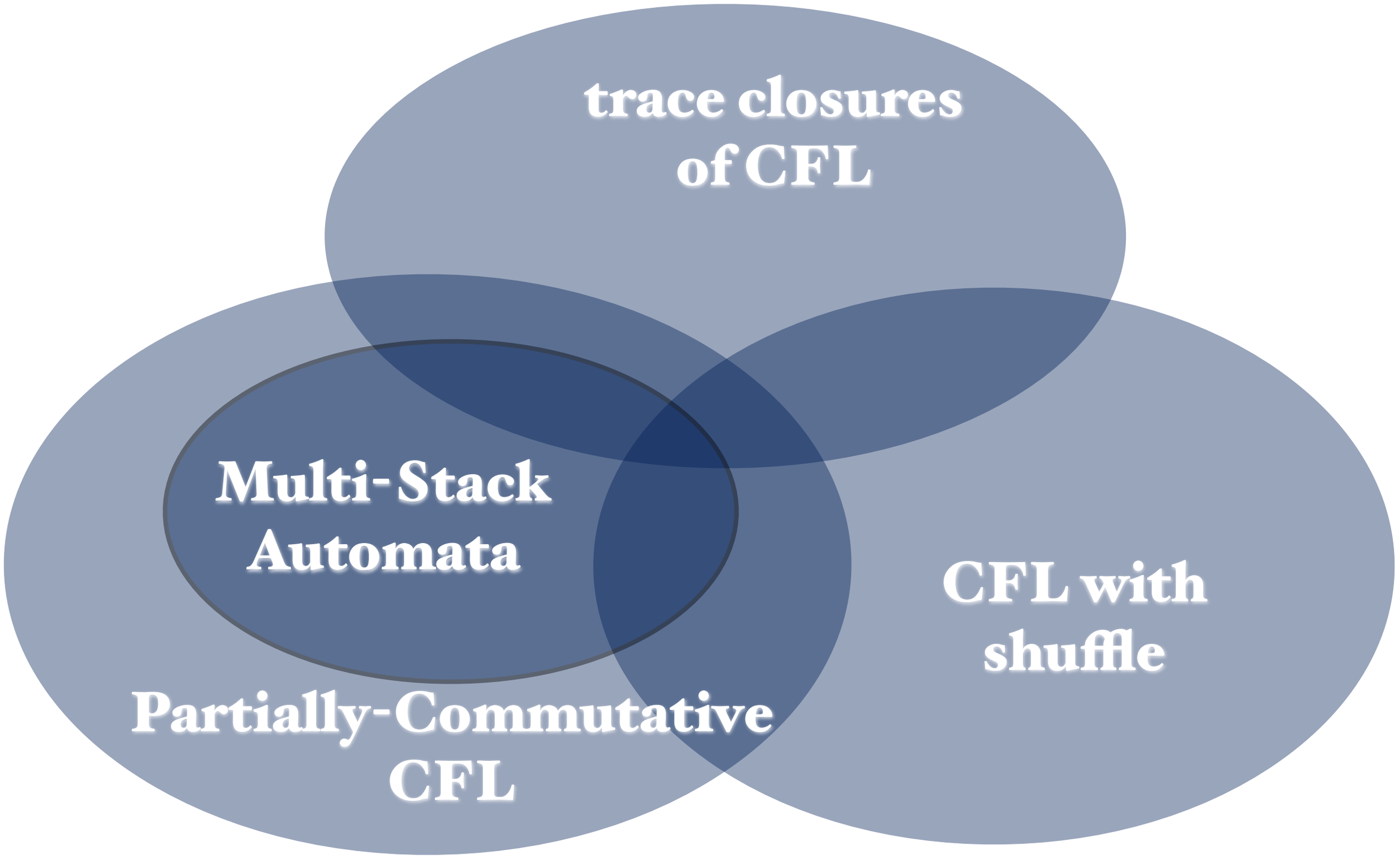
Main result (expressivity)



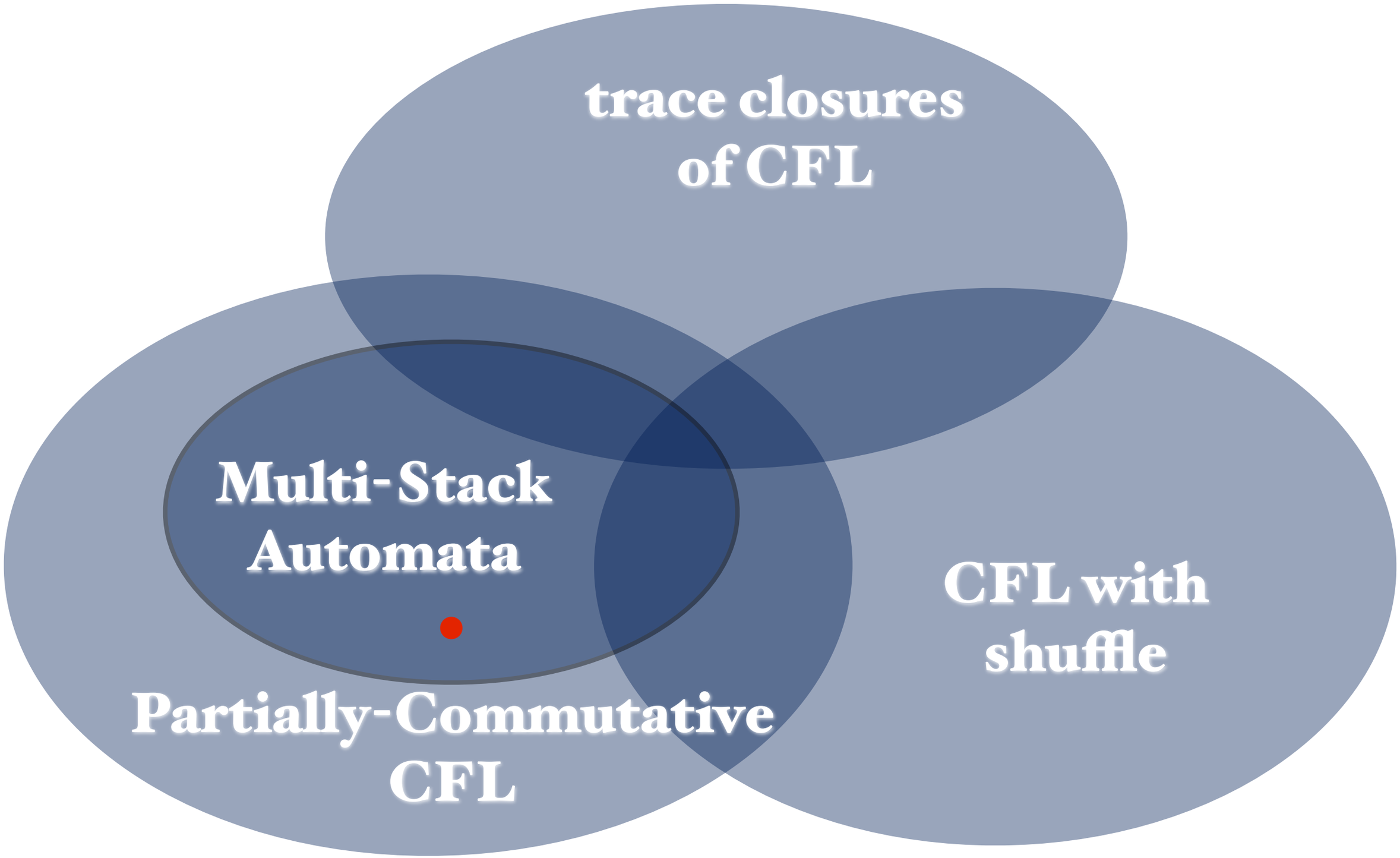
Main result (expressivity)



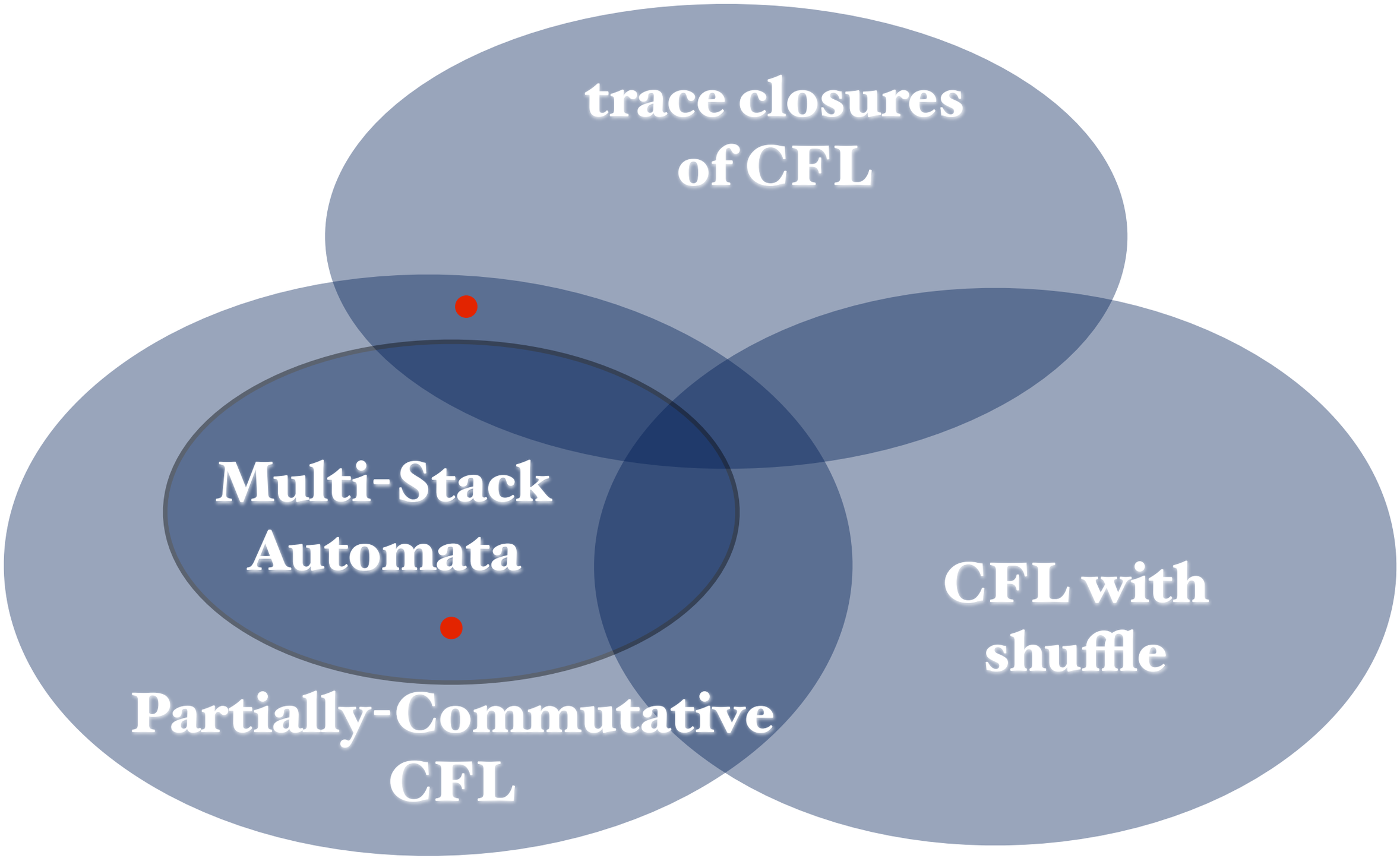
Main result (expressivity)



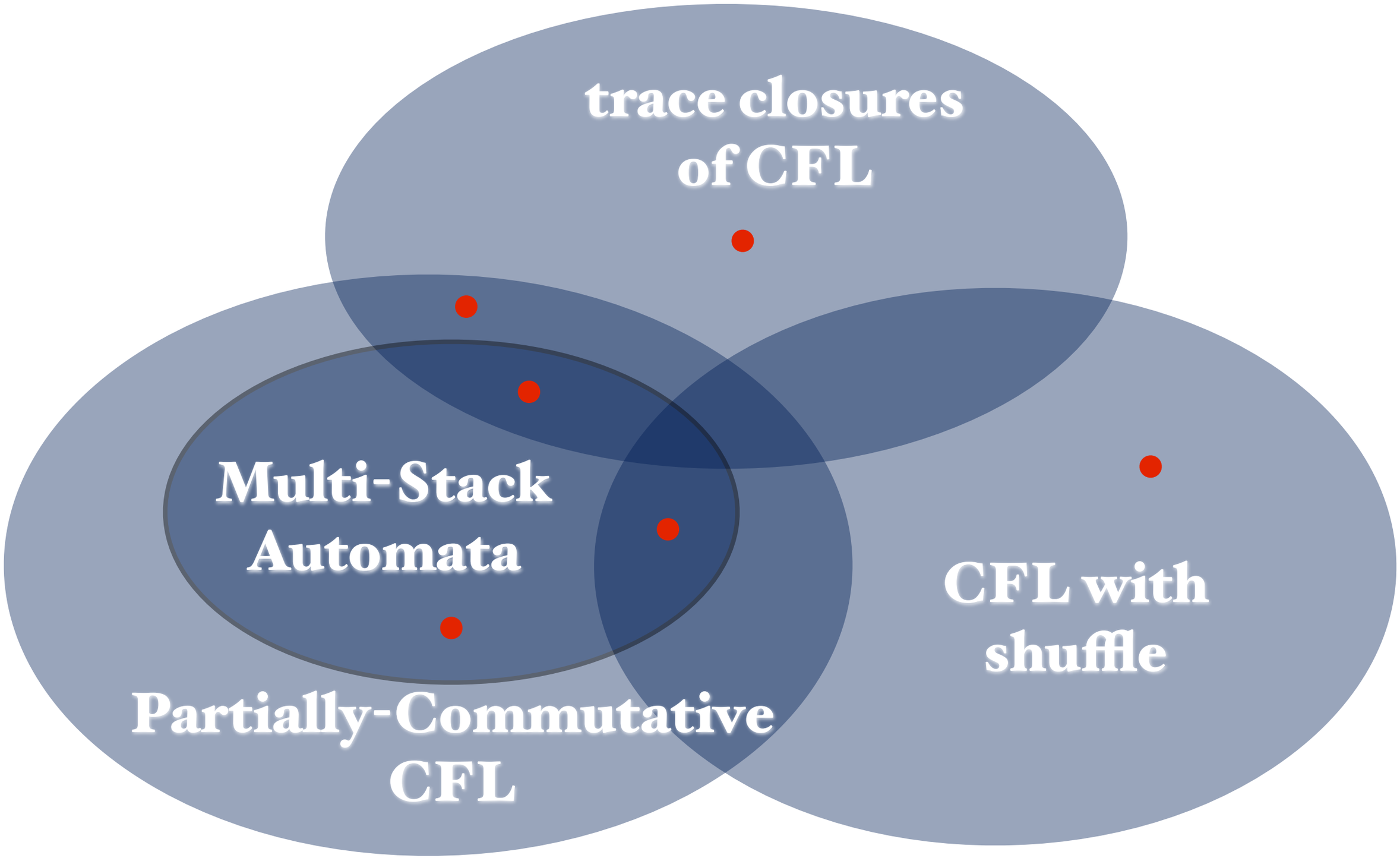
Main result (expressivity)



Main result (expressivity)



Main result (expressivity)



Reachability problem

Reachability problem

Given: A Multi-Stack Automaton and two its configurations: s (source) and t (target)

Reachability problem

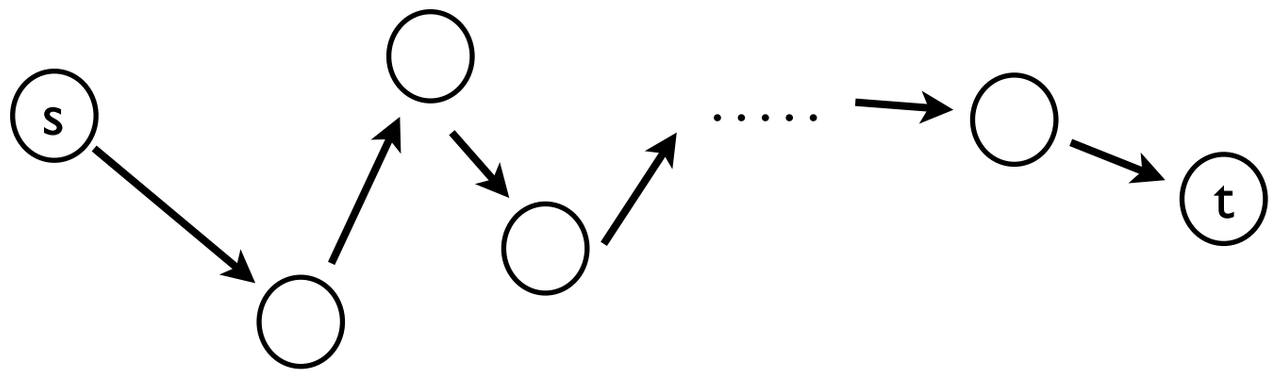
Given: A Multi-Stack Automaton and two its configurations: s (source) and t (target)

Question: Decide whether there is a path from s to t

Reachability problem

Given: A Multi-Stack Automaton and two its configurations: s (source) and t (target)

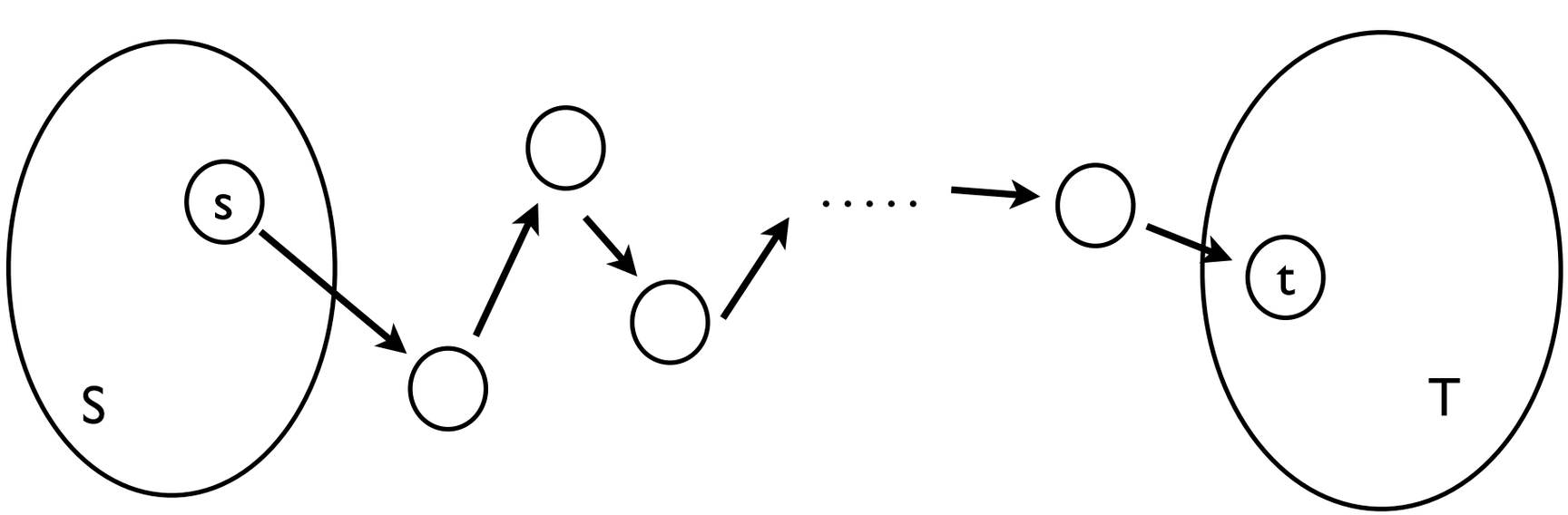
Question: Decide whether there is a path from s to t



Reachability problem

Given: A Multi-Stack Automaton and two its configurations: s (source) and t (target)

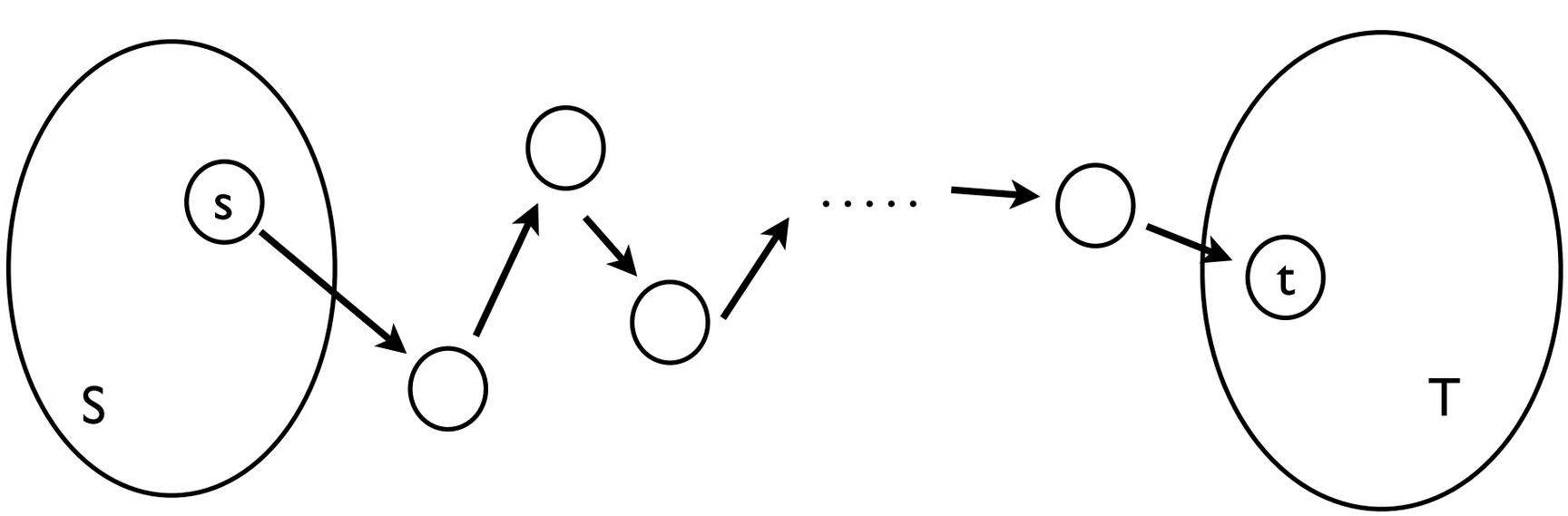
Question: Decide whether there is a path from s to t



Reachability problem

Given: A Multi-Stack Automaton and two its configurations: s (source) and t (target)

Question: Decide whether there is a path from s to t

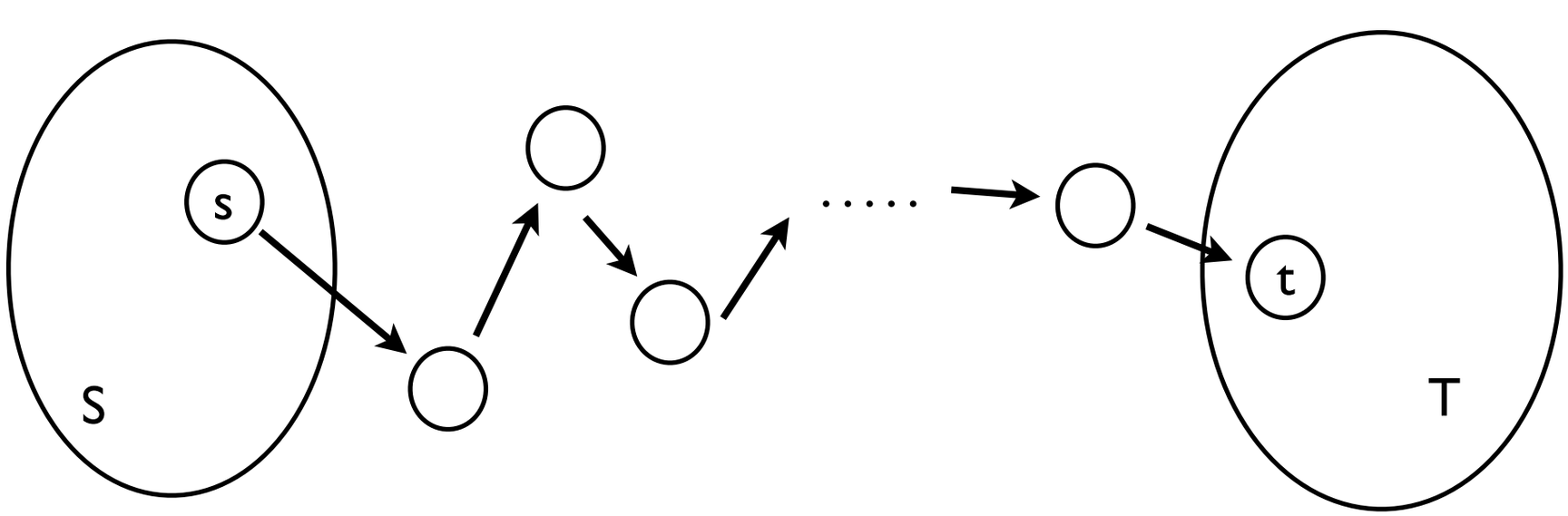


Regular: ((AB)*, CD) ∪ (A*B, C*D*)

Reachability problem

Given: A Multi-Stack Automaton and two its configurations: s (source) and t (target)

Question: Decide whether there is a path from s to t



Regular: ((AB)*, CD) ∪ (A*B, C*D*)

Not regular: (A^n, B^n)

Main result (reachability)

Main result (reachability)

Theorem:
Reachability problem for *normed* MSA with regular source and target sets is NP-complete

Main result (reachability)

Theorem:
Reachability problem for *normed* MSA with regular source and target sets is NP-complete

Normedness: for every letter A

$$\underline{A} \longrightarrow \longrightarrow \cdots \longrightarrow \_$$

Equivalence

Equivalence

Rules are labelled:

Equivalence

Rules are labelled: $A \xrightarrow{a} (B, C)$

Equivalence

Rules are labelled: $A \xrightarrow{a} (B, C)$

The idea: type of the action (c - moving of a car)

Equivalence

Rules are labelled: $A \xrightarrow{a} (B, C)$

The idea: type of the action (c - moving of a car)

Bisimulation equivalence:

Equivalence

Rules are labelled: $A \xrightarrow{a} (B, C)$

The idea: type of the action (c - moving of a car)

Bisimulation equivalence:

s

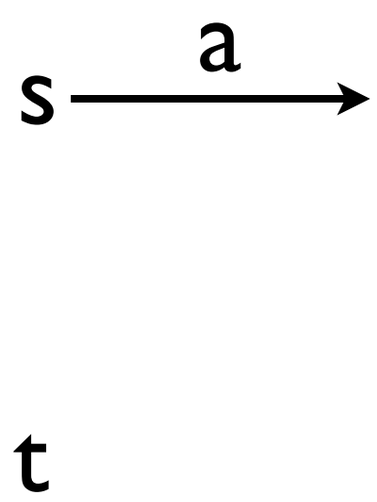
t

Equivalence

Rules are labelled: $A \xrightarrow{a} (B, C)$

The idea: type of the action (c - moving of a car)

Bisimulation equivalence:



Equivalence

Rules are labelled: $A \xrightarrow{a} (B, C)$

The idea: type of the action (c - moving of a car)

Bisimulation equivalence:

$s \xrightarrow{a} s'$

t

Equivalence

Rules are labelled: $A \xrightarrow{a} (B, C)$

The idea: type of the action (c - moving of a car)

Bisimulation equivalence:

$s \xrightarrow{a} s'$

$t \xrightarrow{a}$

Equivalence

Rules are labelled: $A \xrightarrow{a} (B, C)$

The idea: type of the action (c - moving of a car)

Bisimulation equivalence:

$$s \xrightarrow{a} s'$$

$$t \xrightarrow{a} t'$$

Equivalence

Rules are labelled: $A \xrightarrow{a} (B, C)$

The idea: type of the action (c - moving of a car)

Bisimulation equivalence:

$$s \xrightarrow{a} s'$$

$$t \xrightarrow{a} t' \xrightarrow{b}$$

Equivalence

Rules are labelled: $A \xrightarrow{a} (B, C)$

The idea: type of the action (c - moving of a car)

Bisimulation equivalence:

$$s \xrightarrow{a} s'$$

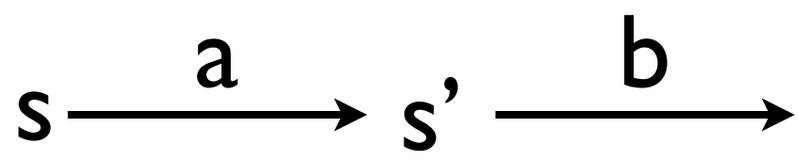
$$t \xrightarrow{a} t' \xrightarrow{b} t''$$

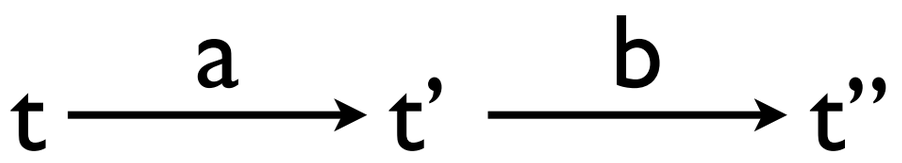
Equivalence

Rules are labelled: $A \xrightarrow{a} (B, C)$

The idea: type of the action (c - moving of a car)

Bisimulation equivalence:



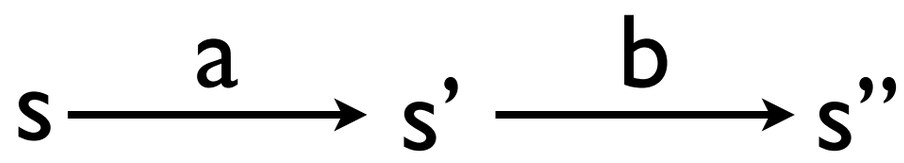


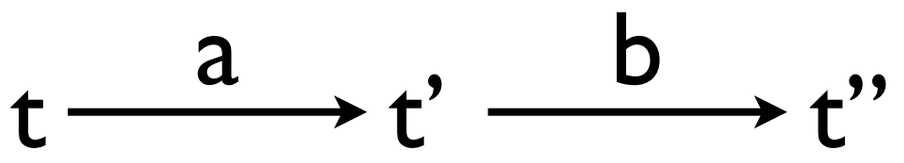
Equivalence

Rules are labelled: $A \xrightarrow{a} (B, C)$

The idea: type of the action (c - moving of a car)

Bisimulation equivalence:





Equivalence problem

Equivalence problem

Given: A Multi-Stack Automaton and two its configurations: s and t

Equivalence problem

Given: A Multi-Stack Automaton and two its configurations: s and t

Question: Decide whether s and t are equivalent

Main results (equivalence)

Main results (equivalence)

Bisimulation equivalence is decidable

Main results (equivalence)

Bisimulation equivalence is decidable

- in polynomial time for normed *disjoint* Multi-Stack Automata

Main results (equivalence)

Bisimulation equivalence is decidable

- in polynomial time for normed *disjoint* Multi-Stack Automata
- in time $O(n^4)$ for normed Single-Stack Automata

Main results (equivalence)

Bisimulation equivalence is decidable

- in polynomial time for normed *disjoint* Multi-Stack Automata
- in time $O(n^4)$ for normed Single-Stack Automata
- in time $O(n^3)$ for Simple Grammars

Thank you!