Regular separability of
languages of
# well-structured transition systems

K Narayan Kumar

CMI Chennai

Wojciech Czerwiński
Sławomir Lasota

University of Warsaw

Roland Mayer
Sebastian Muskalla
Prakash Saivasan

TU Braunschweig

Infinity 2018, Prague

Regular separability of
languages of
well-structured transition systems

K Narayan Kumar

CMI Chennai

Wojciech Czerwiński
Sławomir Lasota

University of Warsaw

Roland Mayer
Sebastian Muskalla
Prakash Saivasan

TU Braunschweig

[Mukund, **Kumar**, Radhakrishnan, Sohoni '98]

Infinity 2018, Prague

languages of **finite words**

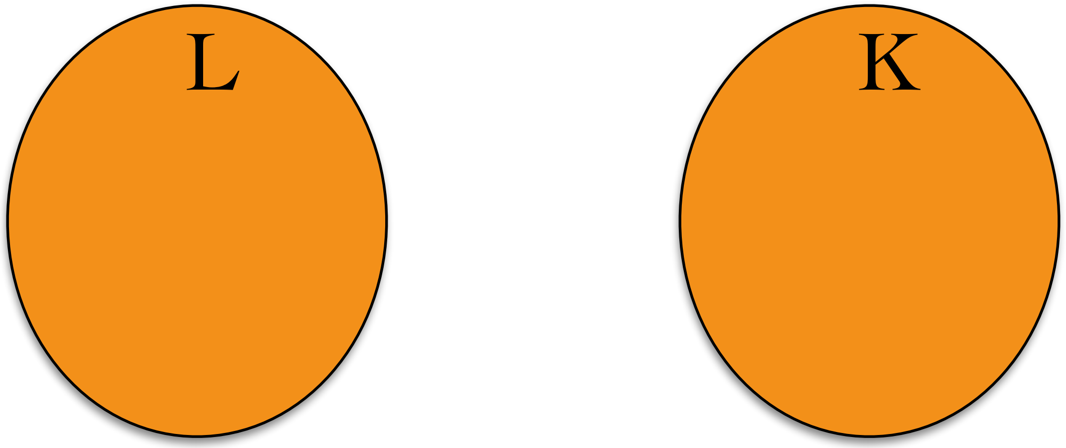
Regular separability

Fix a class of languages C

Regular separability

Fix a class of languages C

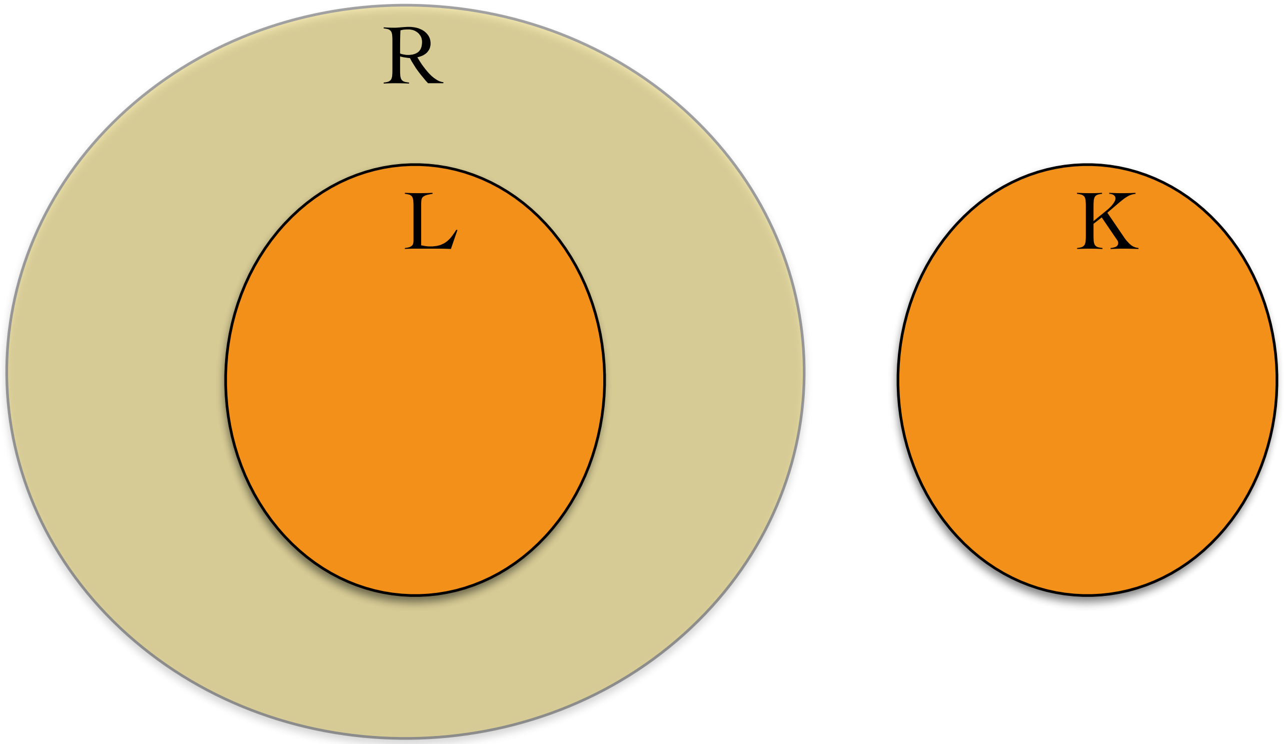
Input: two (disjoint) languages L, K from C



Regular separability

Fix a class of languages *C*

Input: two (disjoint) languages L, K from *C*

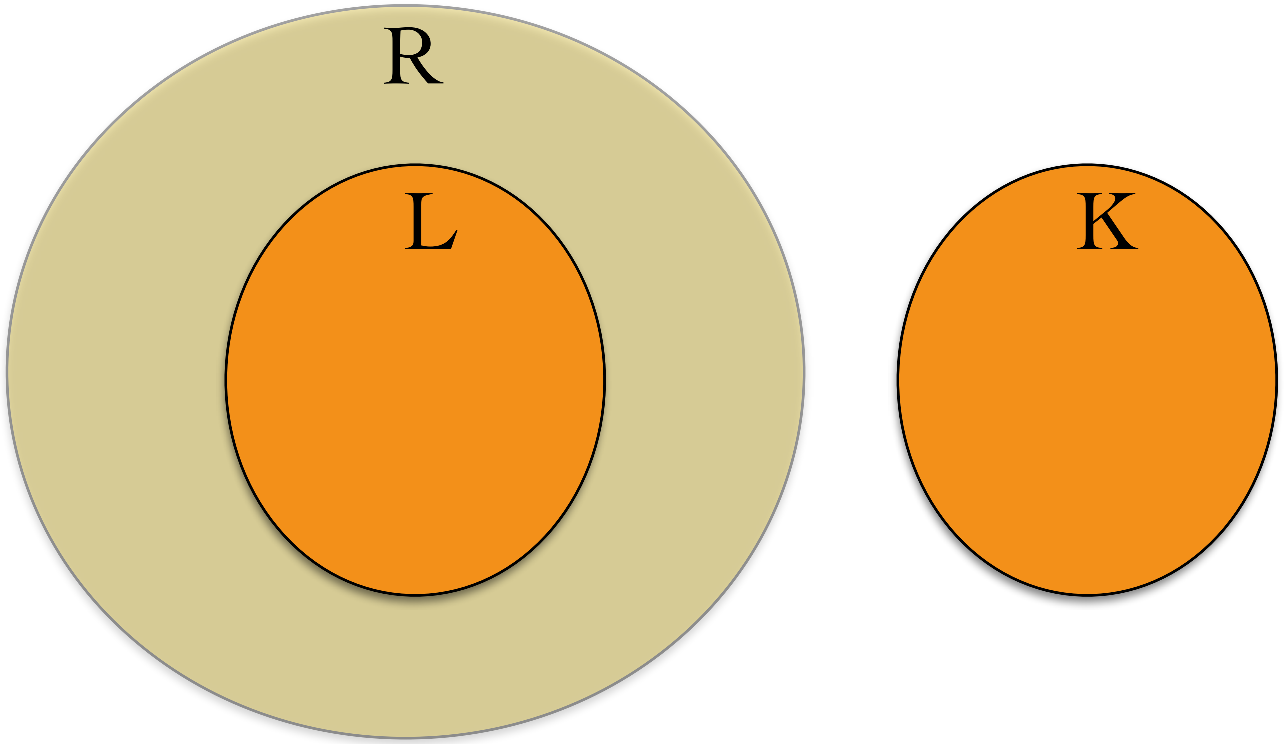


Question: are these two languages **separated** by a regular language?

Regular separability

Fix a class of languages C

Input: two (disjoint) languages L, K from C

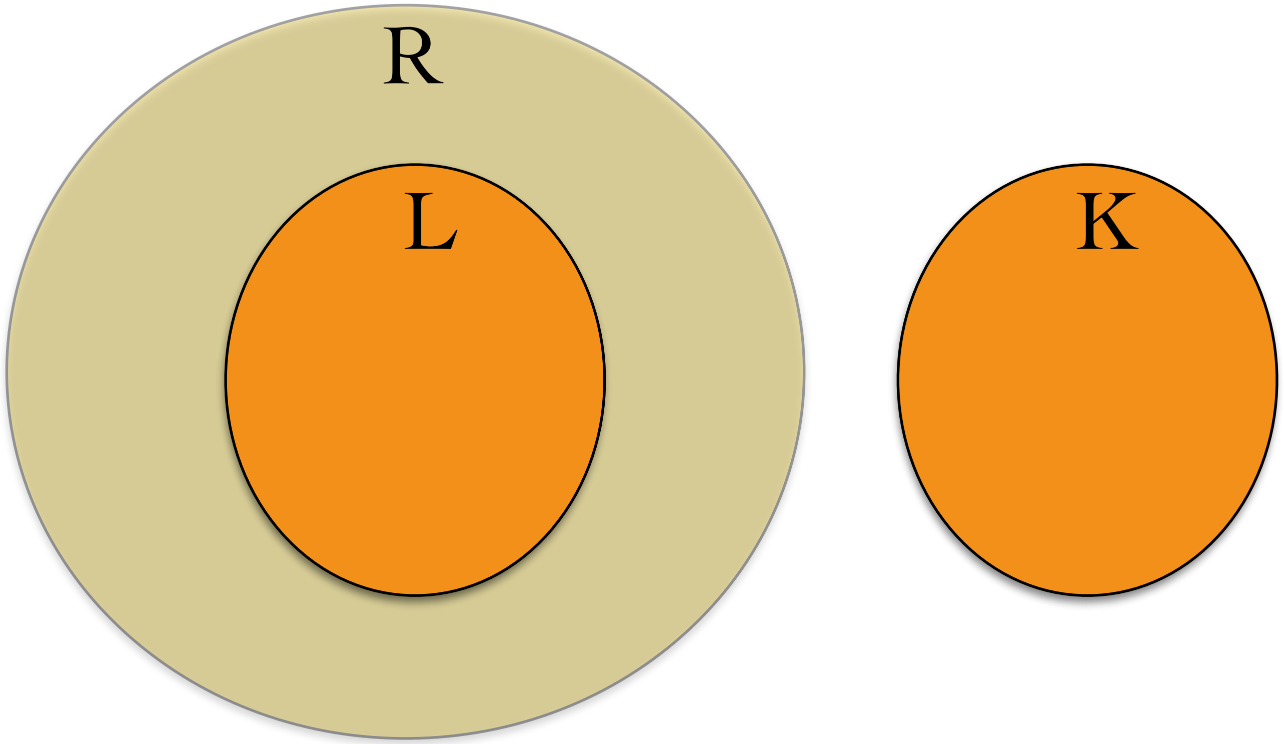


Question: are these two languages **separated** by a regular language?
I.e., is there a regular language R with $L \subseteq R$ and $R \cap K = \emptyset$?

Regular separability

Fix a class of languages C

Input: two (disjoint) languages L, K from C



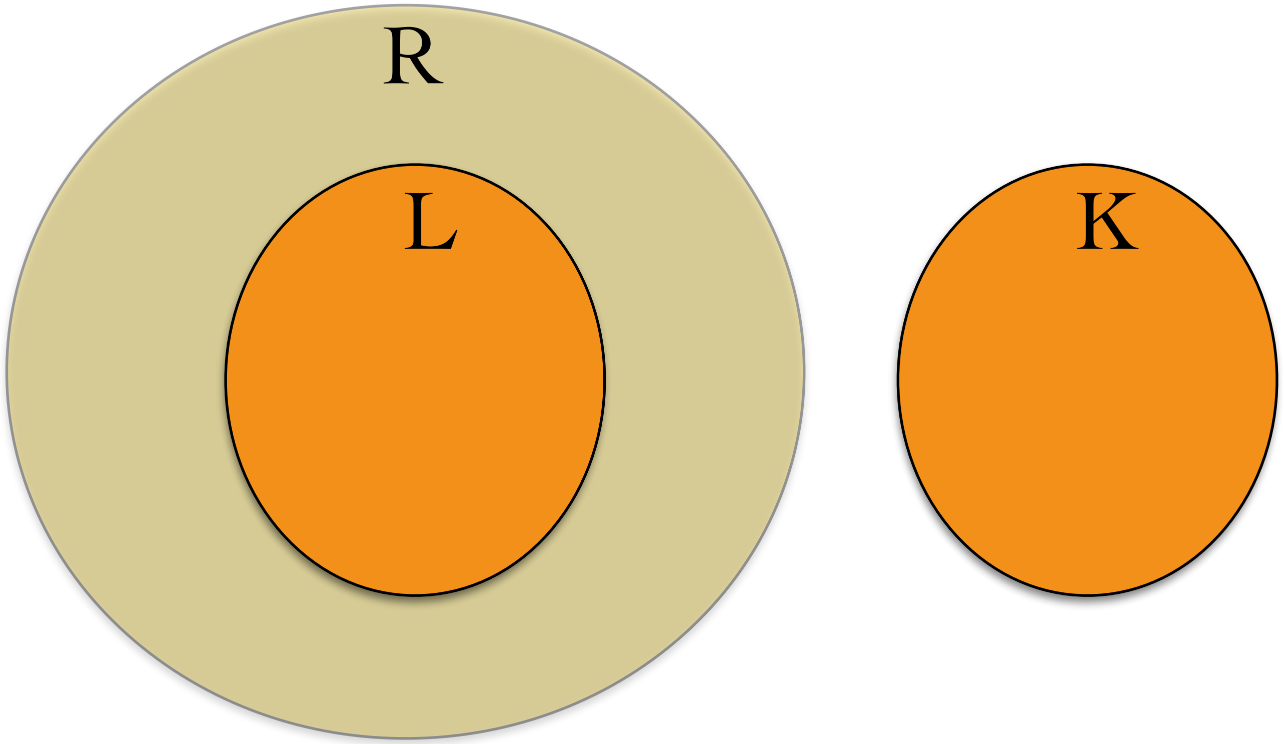
Symmetric in L, K

Question: are these two languages **separated** by a regular language?
I.e., is there a regular language R with $L \subseteq R$ and $R \cap K = \varnothing$?

Regular separability

Fix a class of languages $\mathcal{C}$

Input: two (disjoint) languages L, K from $\mathcal{C}$



Symmetric in L, K

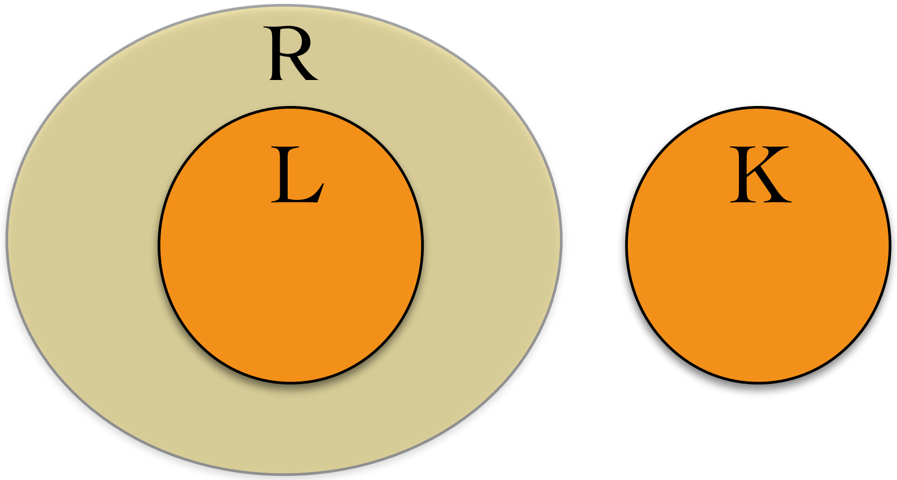
Parametric in $\mathcal{C}$

Question: are these two languages **separated** by a regular language?
I.e., is there a regular language R with $L \subseteq R$ and $R \cap K = \emptyset$?

Is regular separability useful?

Is regular separability useful?

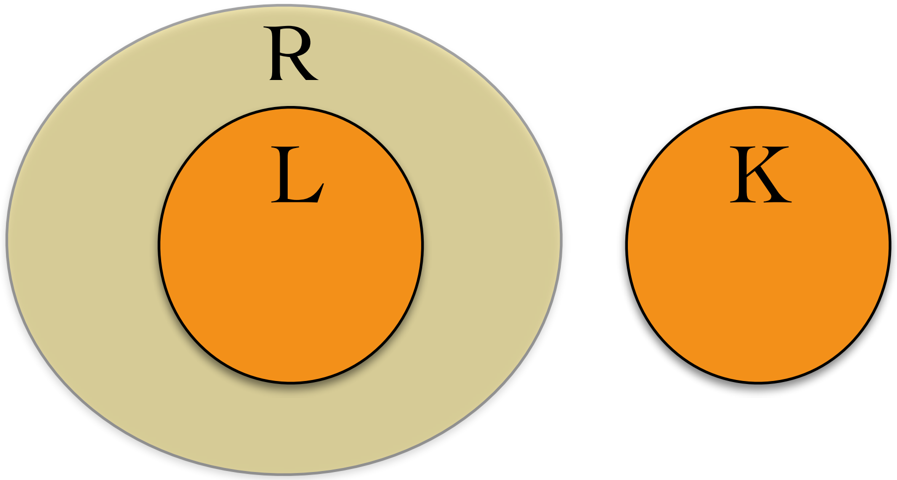
separator as a classifier:



classify a word from L ∪ K
into L or K

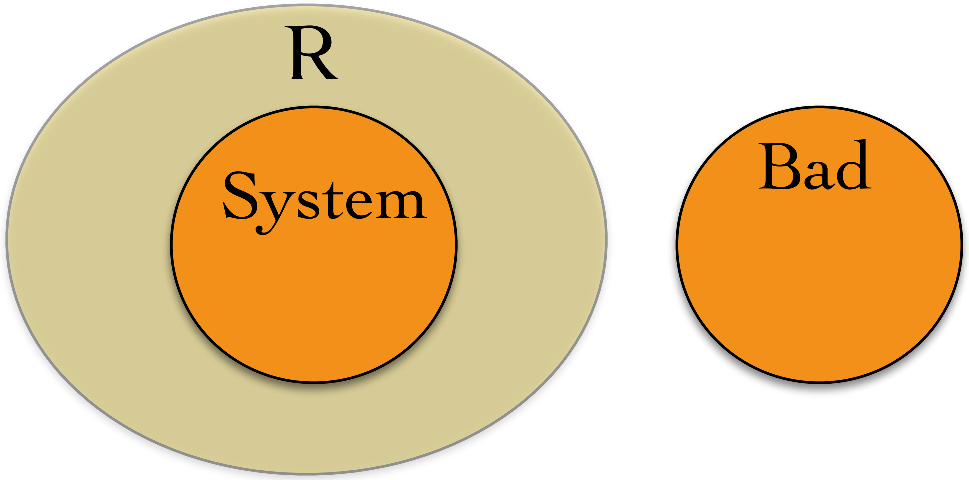
Is regular separability useful?

separator as a classifier:



classify a word from L ∪ K into L or K

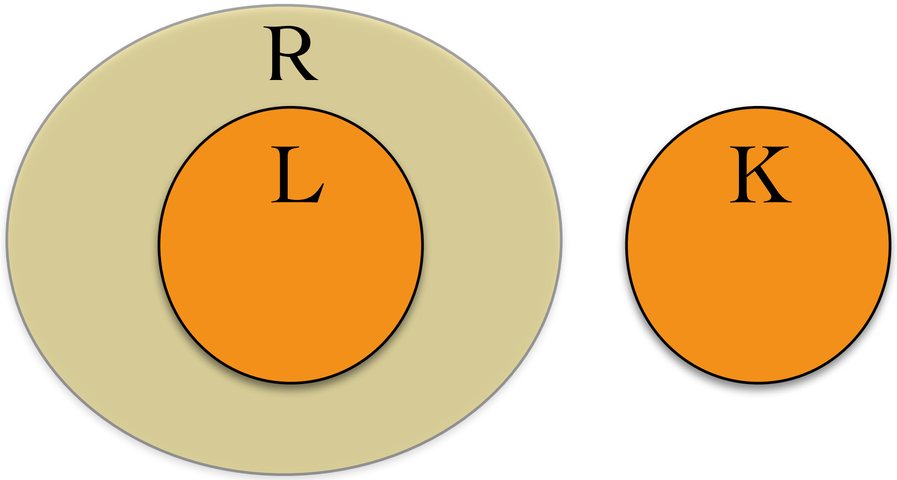
language-theoretic verification:



separator proves absence of undesirable behavior

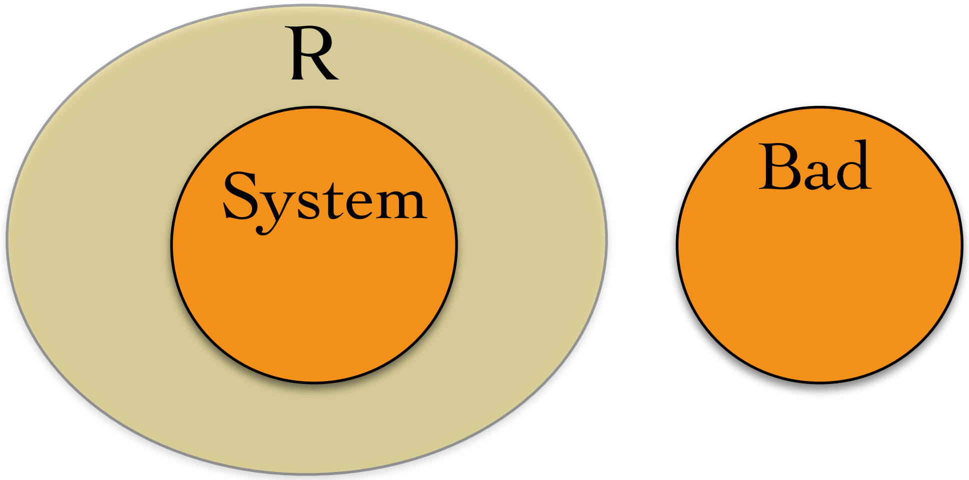
Is regular separability useful?

separator as a classifier:



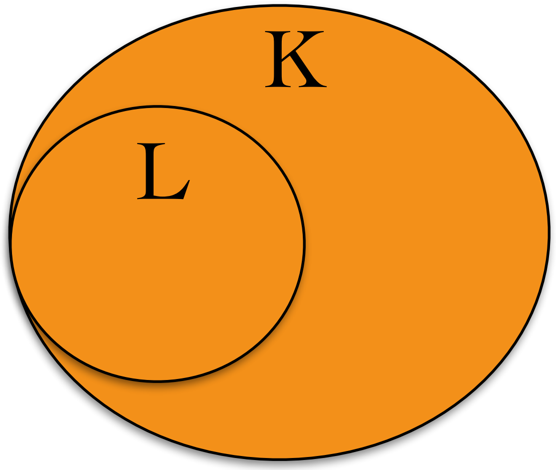
classify a word from L ∪ K into L or K

language-theoretic verification:



separator proves absence of undesirable behavior

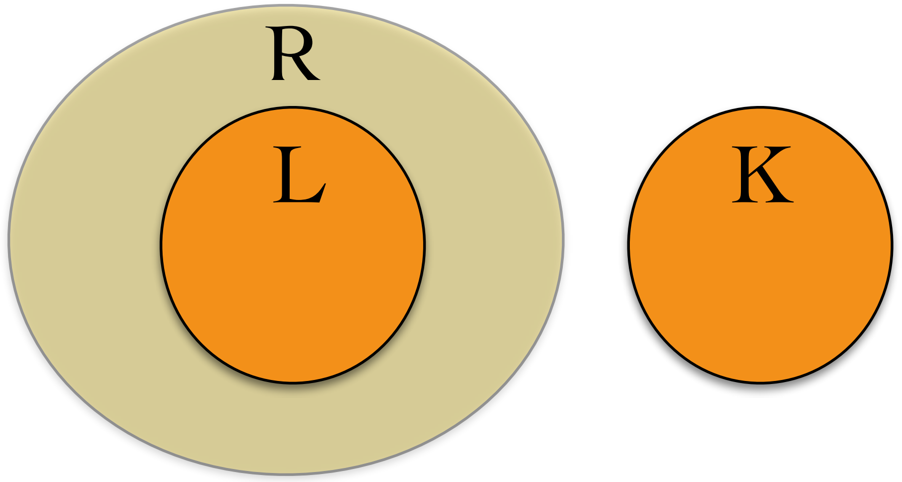
separator as a recognizer:



recognize L inside K

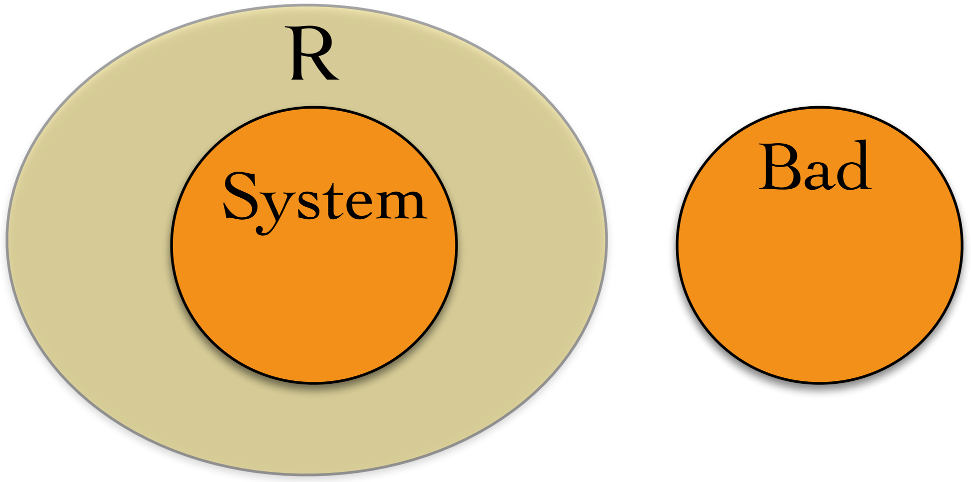
Is regular separability useful?

separator as a classifier:



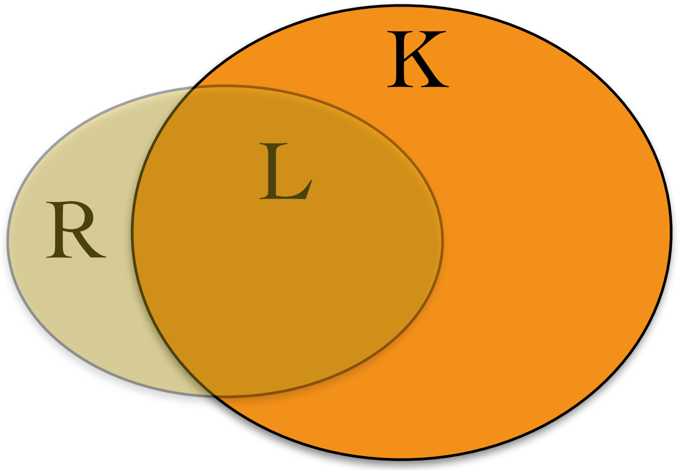
classify a word from L ∪ K into L or K

language-theoretic verification:



separator proves absence of undesirable behavior

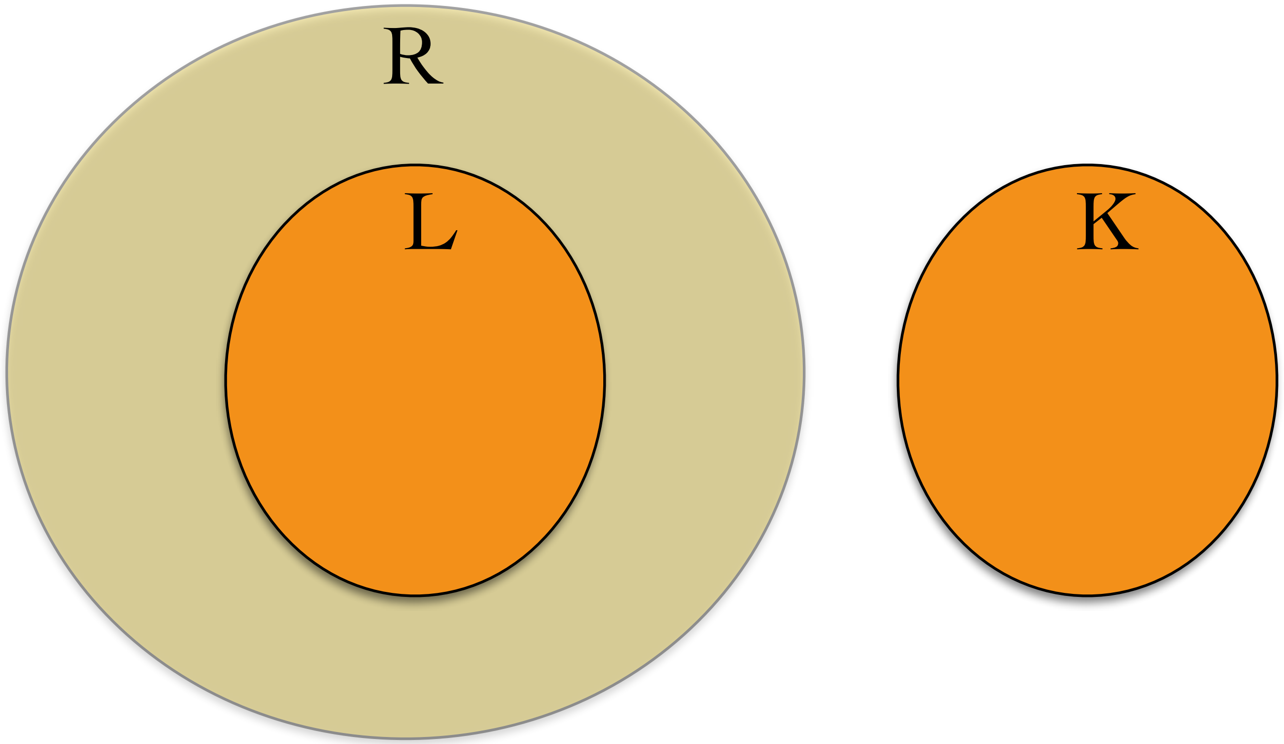
separator as a recognizer:



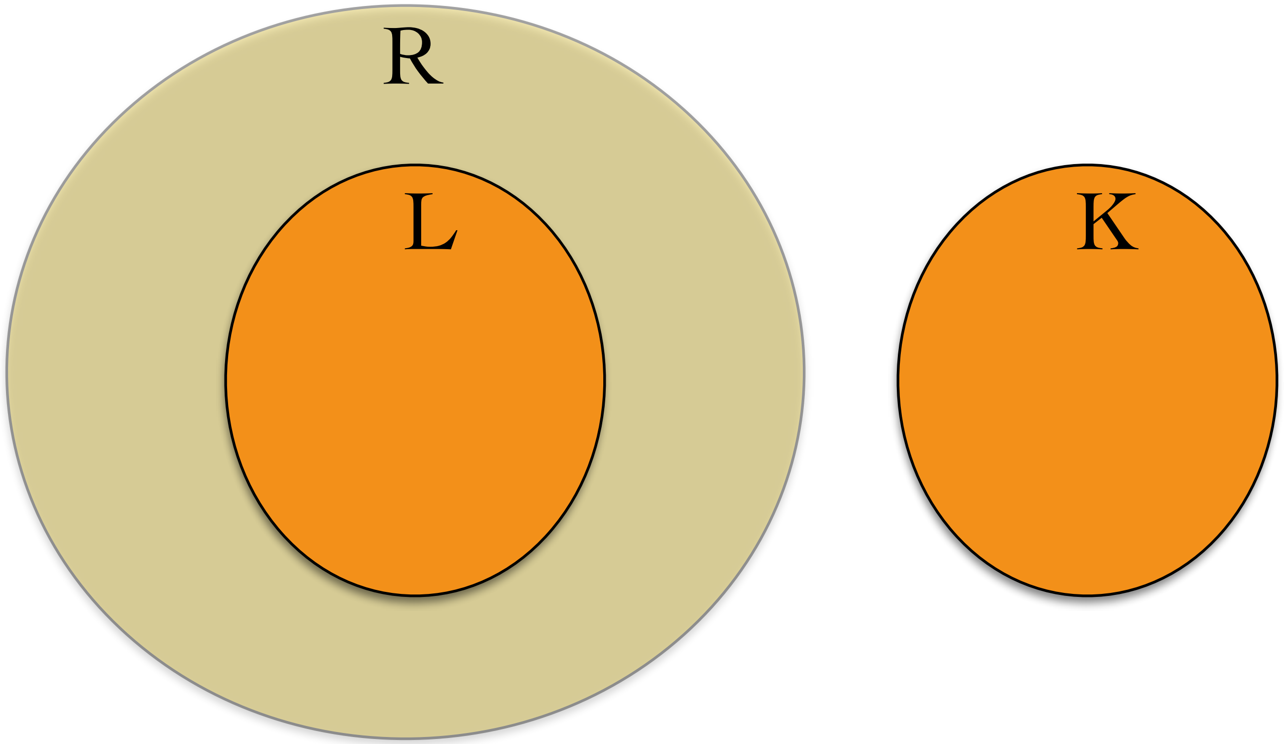
recognize L inside K

Regular separability

Regular separability

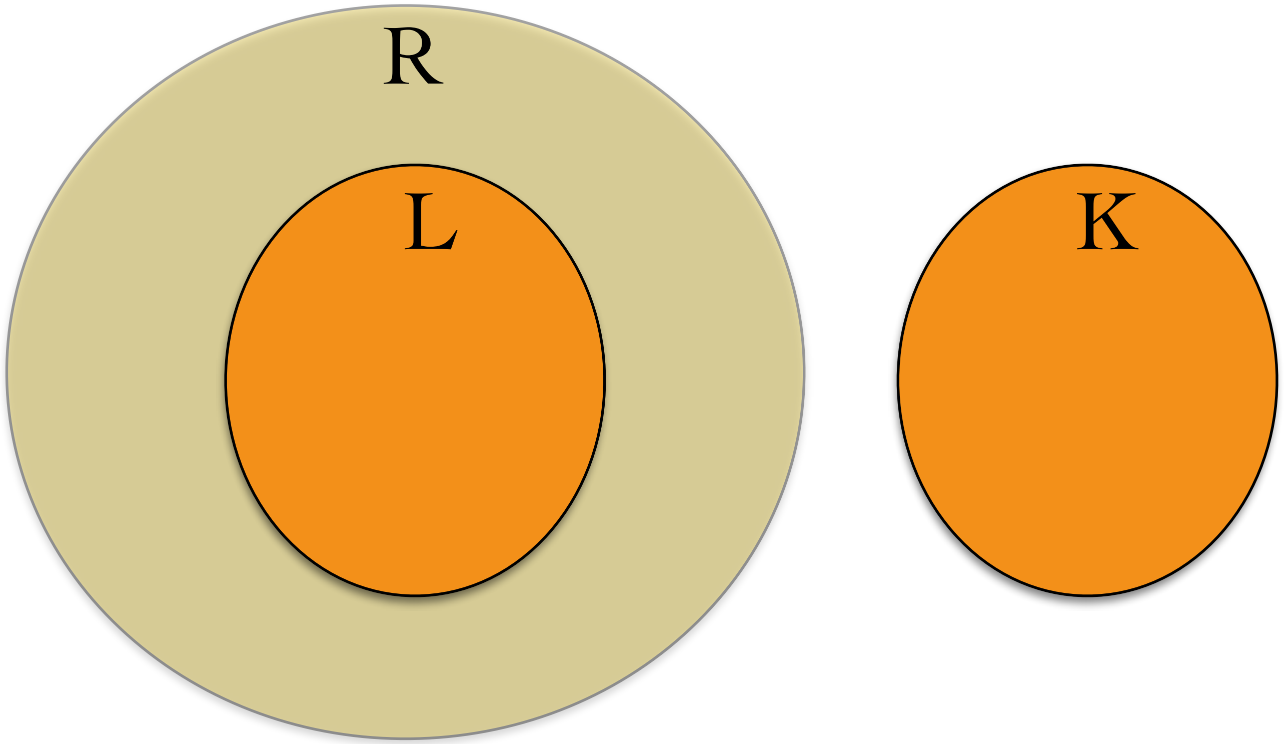


Regular separability



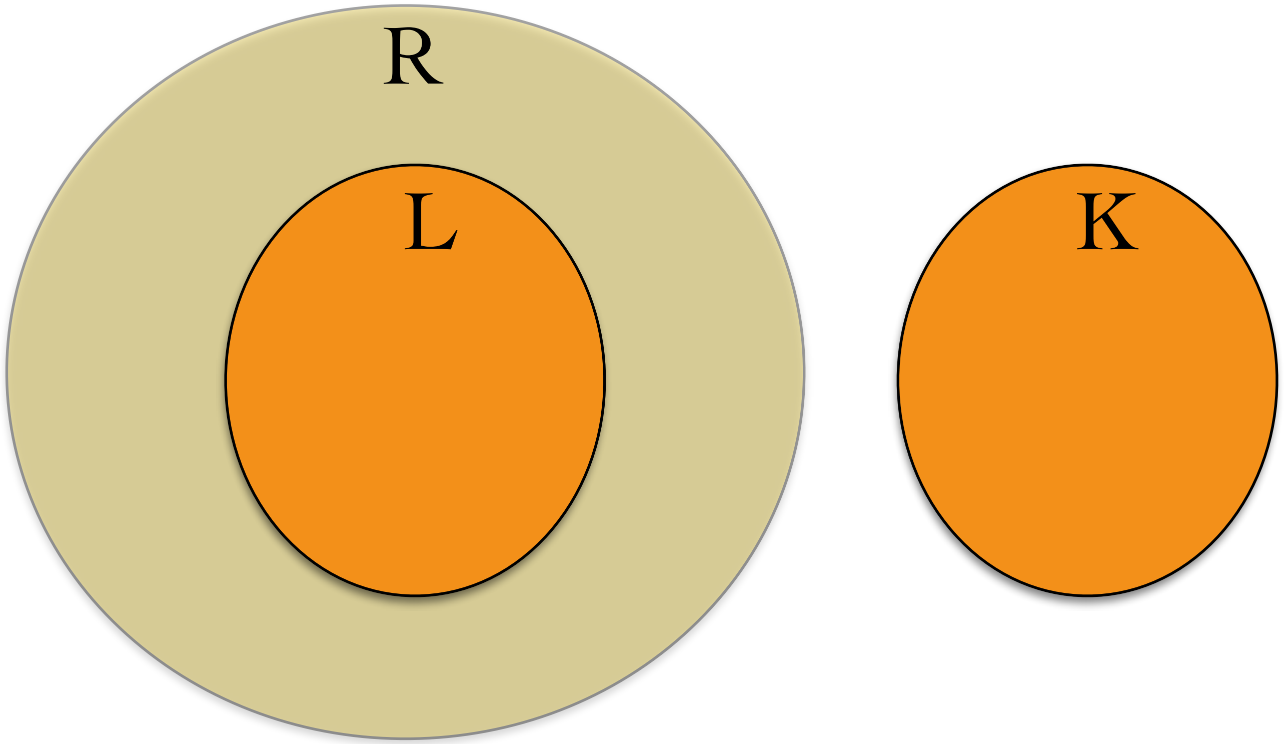
- **decision problem**: are given L, K regular-separable?

Regular separability



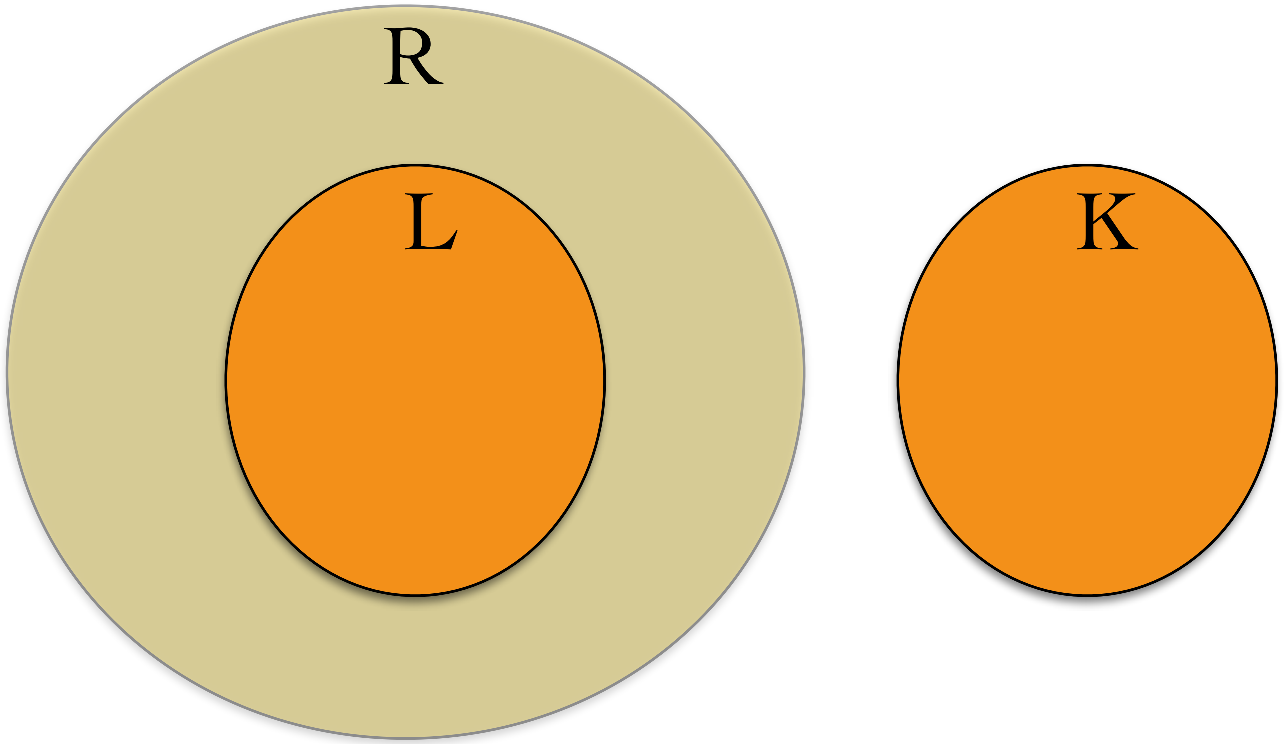
- **decision problem**: are given L, K regular-separable?
- **computation problem**: compute a regular separator of given L, K

Regular separability



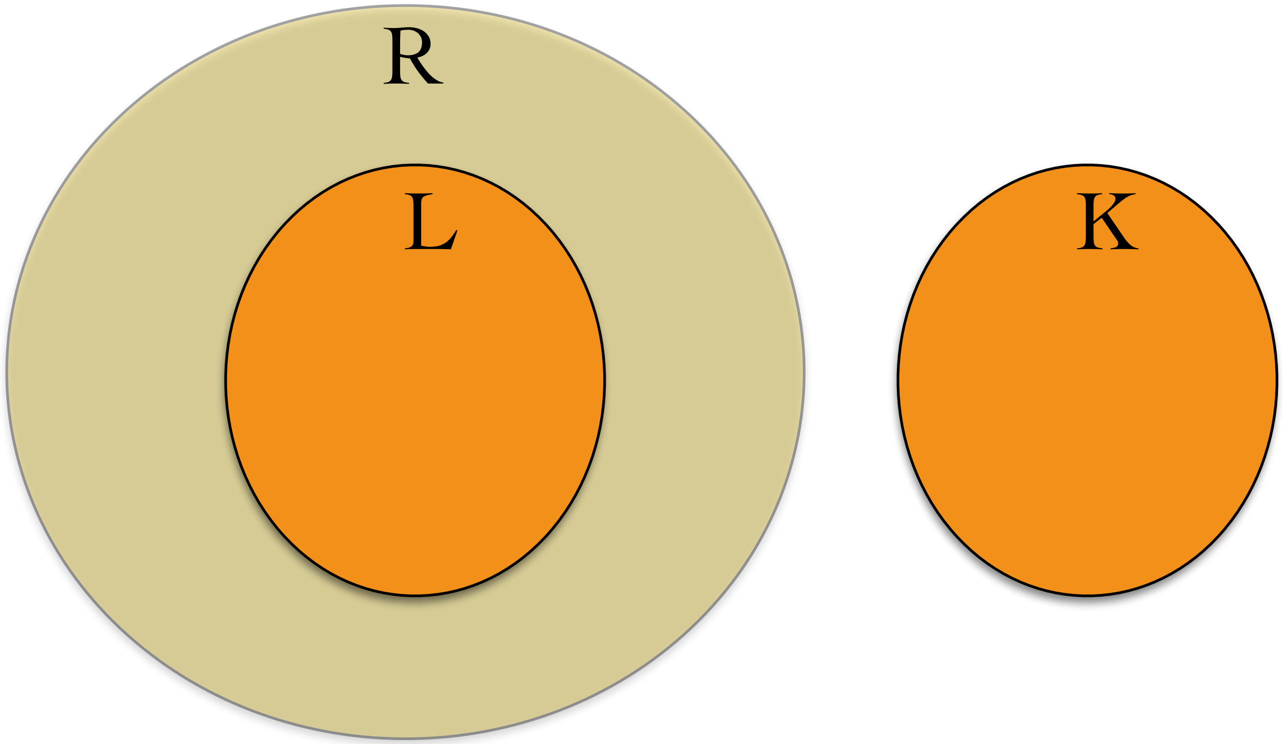
- **decision problem**: are given L, K regular-separable?
- **computation problem**: compute a regular separator of given L, K
- **qualitative characterization**: sufficient (and necessary) condition for regular-separability

Regular separability



- **decision problem**: are given L, K regular-separable?
- **computation problem**: compute a regular separator of given L, K
- **qualitative characterization**: sufficient (and necessary) condition for regular-separability
- **quantitative characterization**: bound on the size of a separator

Regular separability



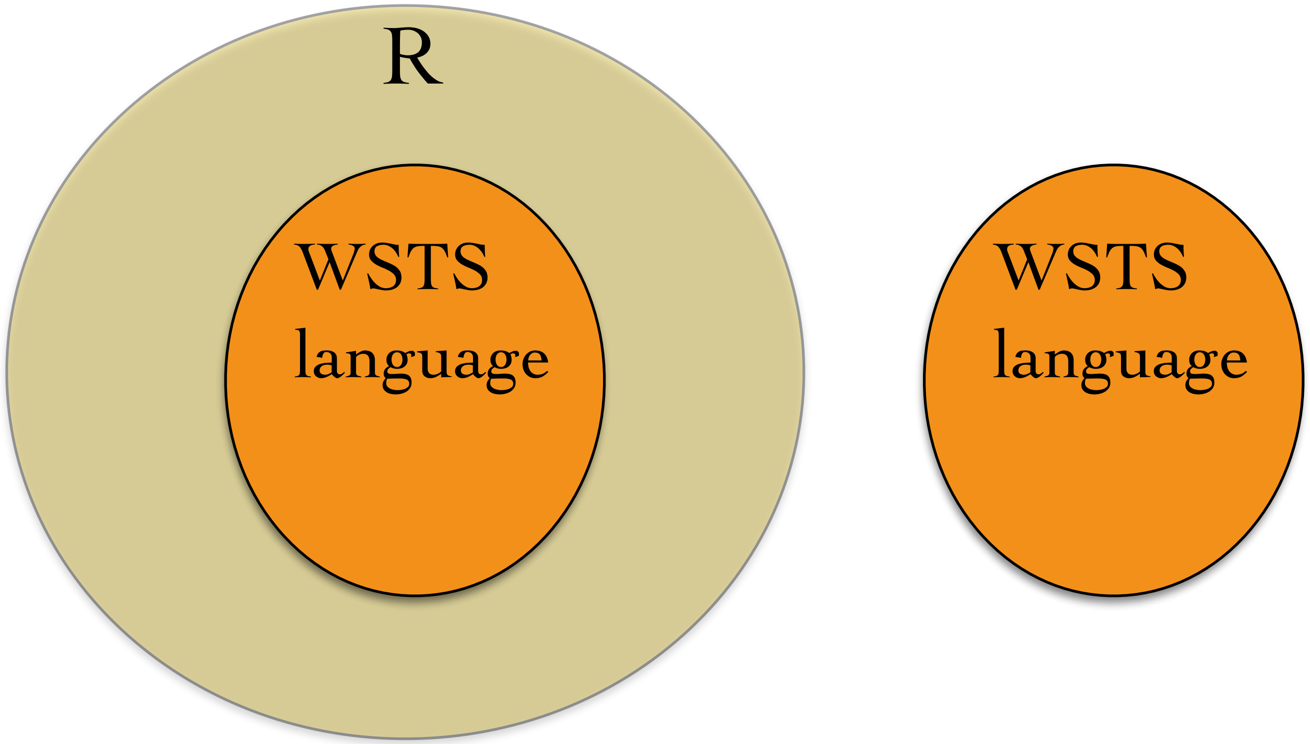
- **decision problem**: are given L, K regular-separable?

- **computation problem**: compute a regular separator of given L, K

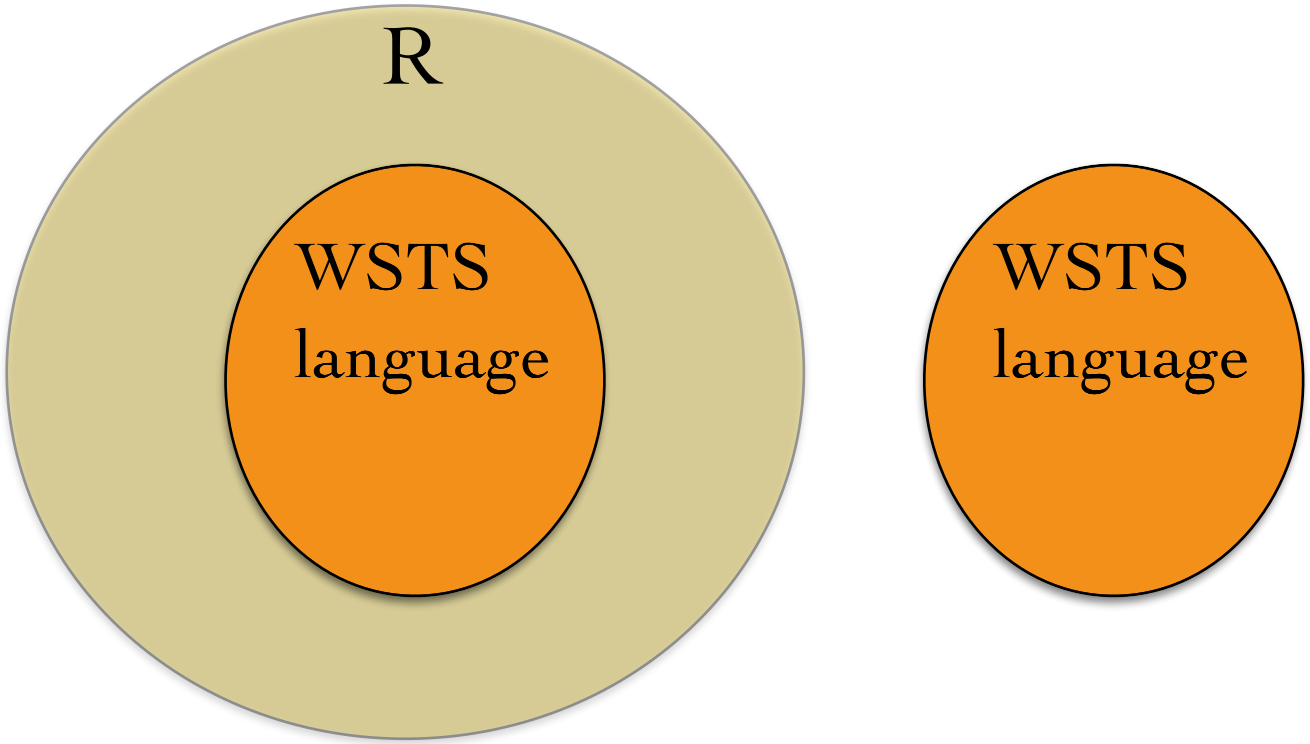
- **qualitative characterization**: sufficient (and necessary) condition for regular-separability

- **quantitative characterization**: bound on the size of a separator

Regular separability of WSTS languages

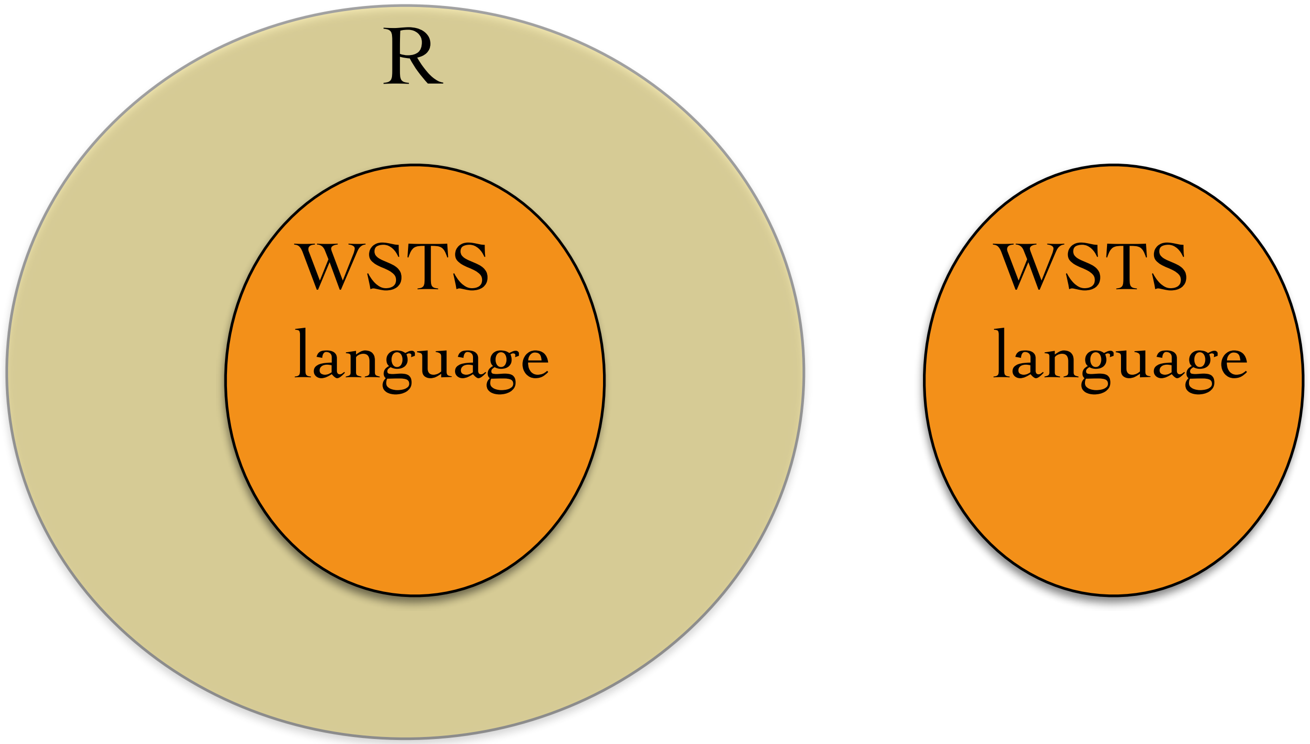


Regular separability of WSTS languages



Theorem: Every two disjoint WSTS languages are regular-separable,

Regular separability of WSTS languages



Theorem: Every two disjoint WSTS languages are regular-separable, under some mild assumptions.

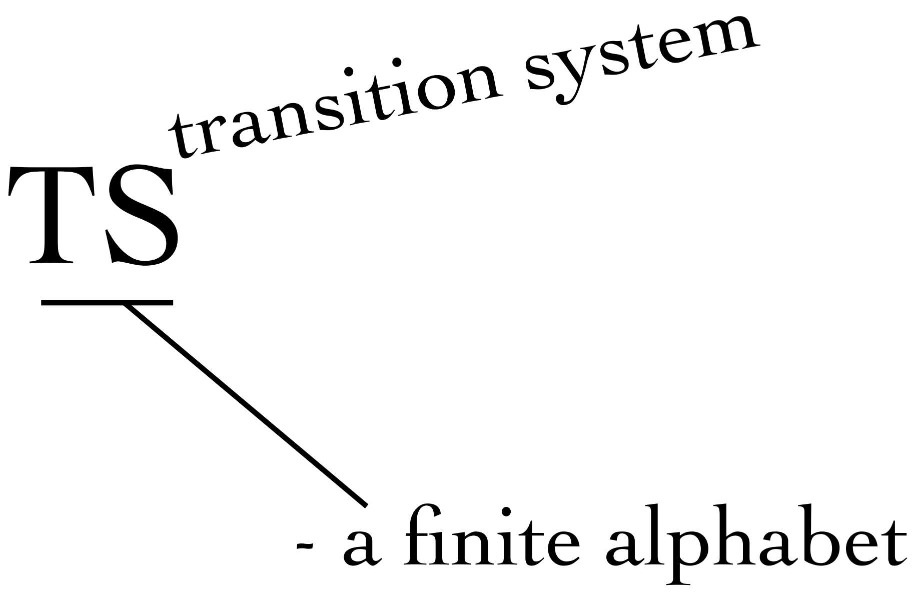
$^{U}/_{D}$WSTS: well-structured transition system

$^{\mathrm{U}}/_{\mathrm{D}}$WSTS: well-structured transition system

transition system

TS

$^{U}/_{D}$WSTS: well-structured transition system

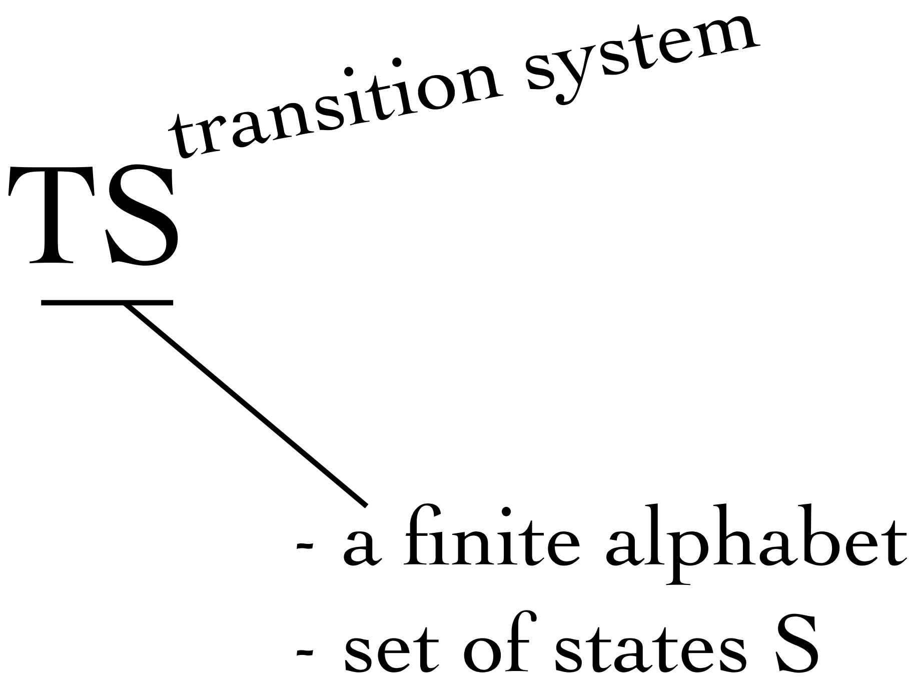


TS

transition system

- a finite alphabet

$^{U}/_{D}$WSTS: well-structured transition system



TS

transition system

- a finite alphabet
- set of states S

$^{U}/_{D}$WSTS: well-structured transition system

TS

transition system

- a finite alphabet
- set of states S
- a subset I of initial states

$^{U}/_{D}$WSTS: well-structured transition system

transition system

TS

- a finite alphabet
- set of states S
- a subset I of initial states
- a subset F of final states

$^{U}/_{D}$WSTS: well-structured transition system

TS transition system

- a finite alphabet
- set of states S
- a subset I of initial states
- a subset F of final states
- transition relation $\xrightarrow{a}$

the language of a TS

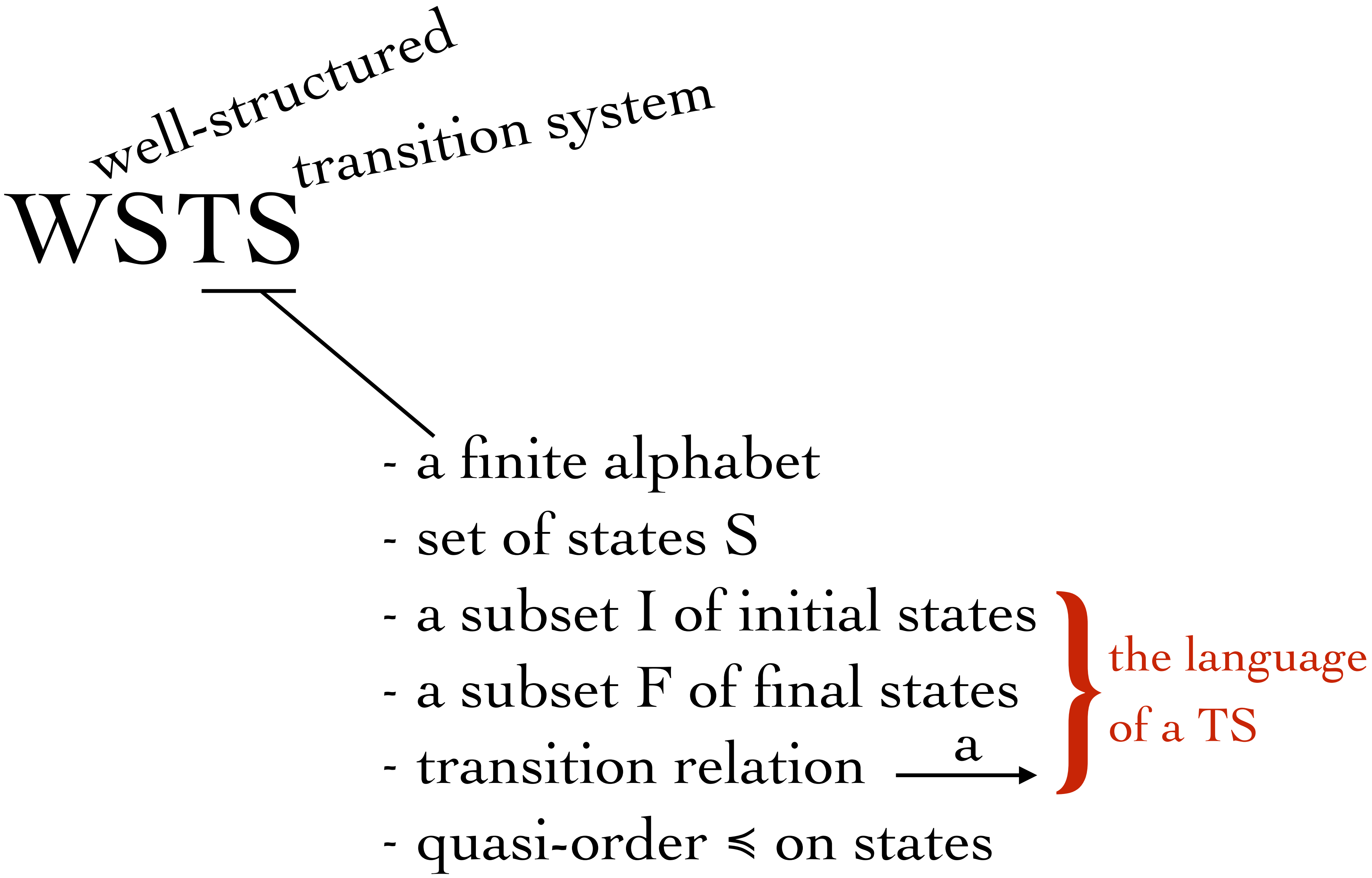
${}^{U}/_{D}$WSTS: well-structured transition system

TS transition system

- a finite alphabet
- set of states S
- a subset I of initial states
- a subset F of final states
- transition relation $\xrightarrow{a}$
- quasi-order ≼ on states

the language of a TS

$^{U}/_{D}$WSTS: well-structured transition system



well-structured

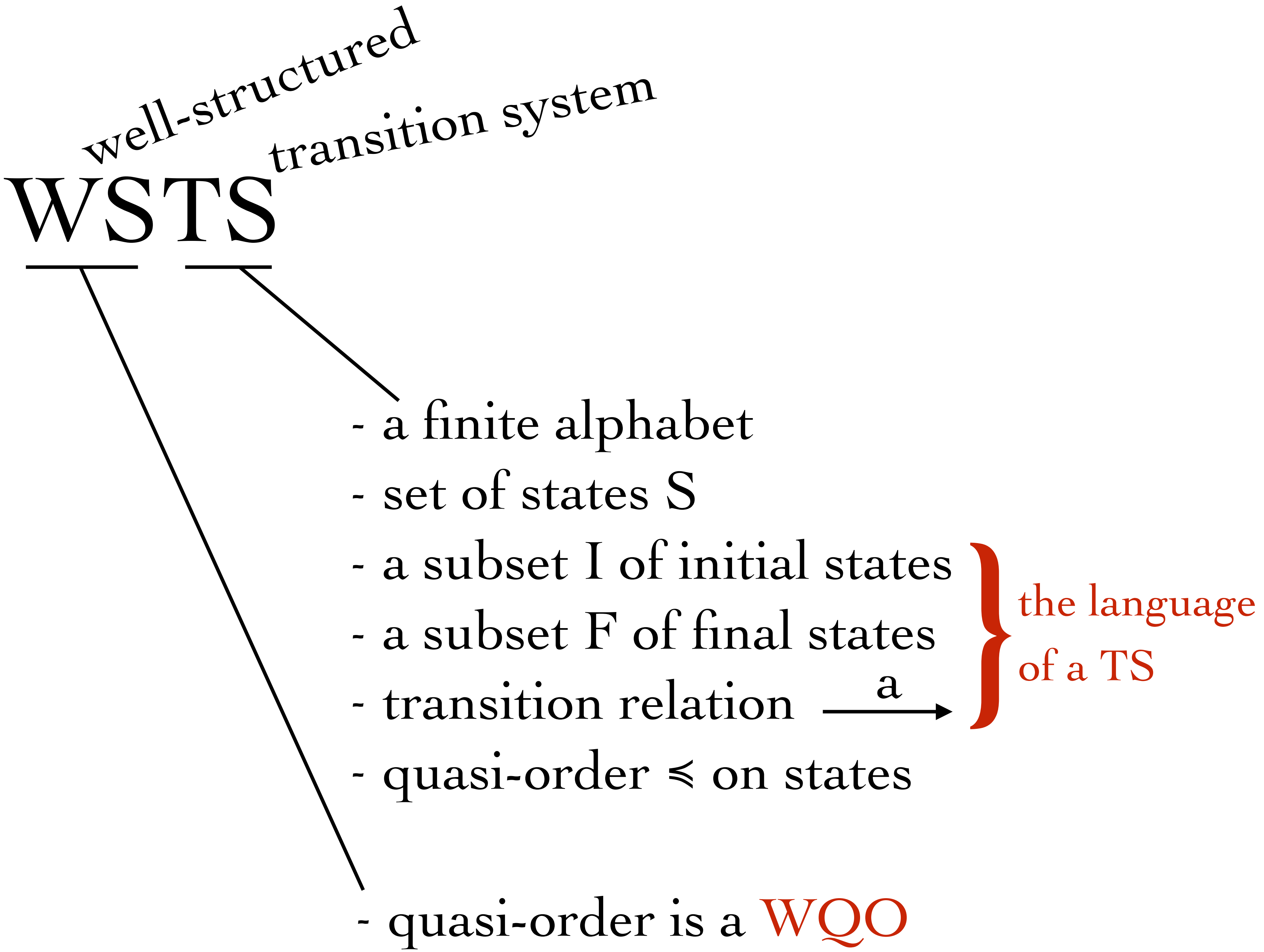
transition system

WSTS

- a finite alphabet
- set of states S
- a subset I of initial states
- a subset F of final states
- transition relation $\xrightarrow{a}$
- quasi-order $\preccurlyeq$ on states

the language of a TS

$^{U}/_{D}$WSTS: well-structured transition system



well-structured

transition system

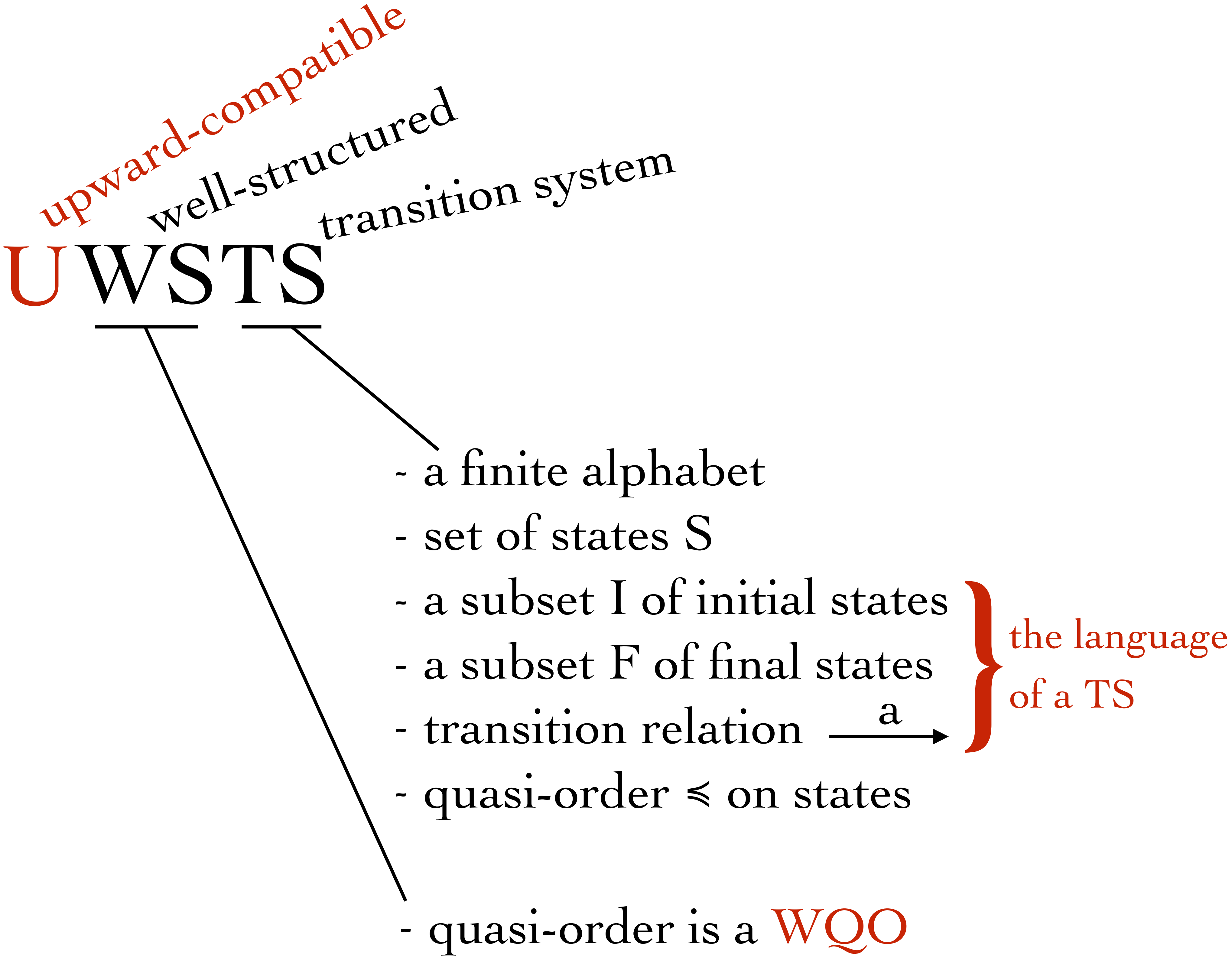
WSTS

- a finite alphabet
- set of states S
- a subset I of initial states
- a subset F of final states
- transition relation $\xrightarrow{a}$
- quasi-order ≼ on states

the language of a TS

- quasi-order is a WQO

$^{U}/_{D}$WSTS: well-structured transition system



upward-compatible

well-structured

transition system

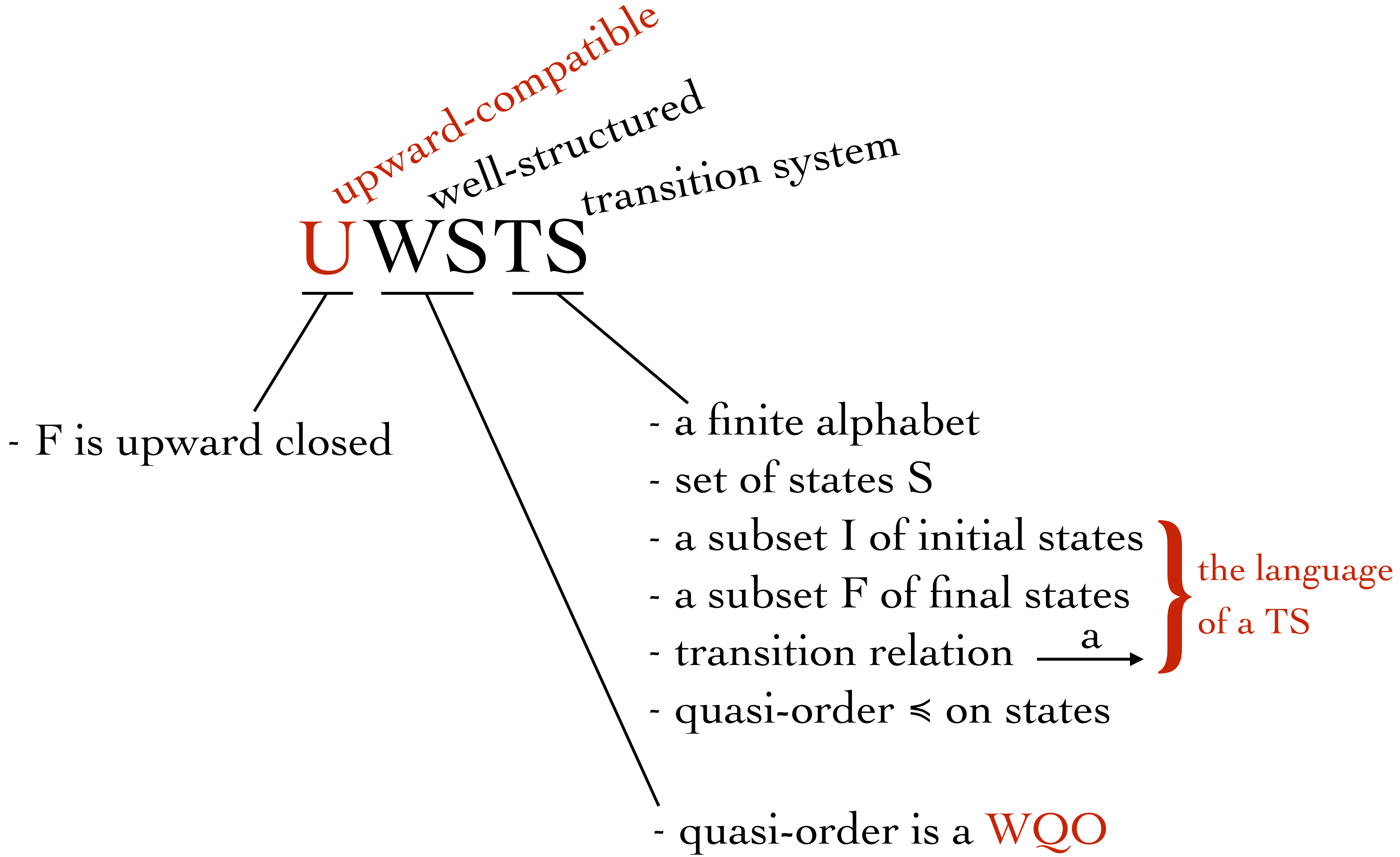
UWSTS

- a finite alphabet
- set of states S
- a subset I of initial states
- a subset F of final states
- transition relation $\xrightarrow{a}$
- quasi-order $\preccurlyeq$ on states

the language of a TS

- quasi-order is a WQO

$^{U}/_{D}$WSTS: well-structured transition system



upward-compatible

well-structured

transition system

UWSTS

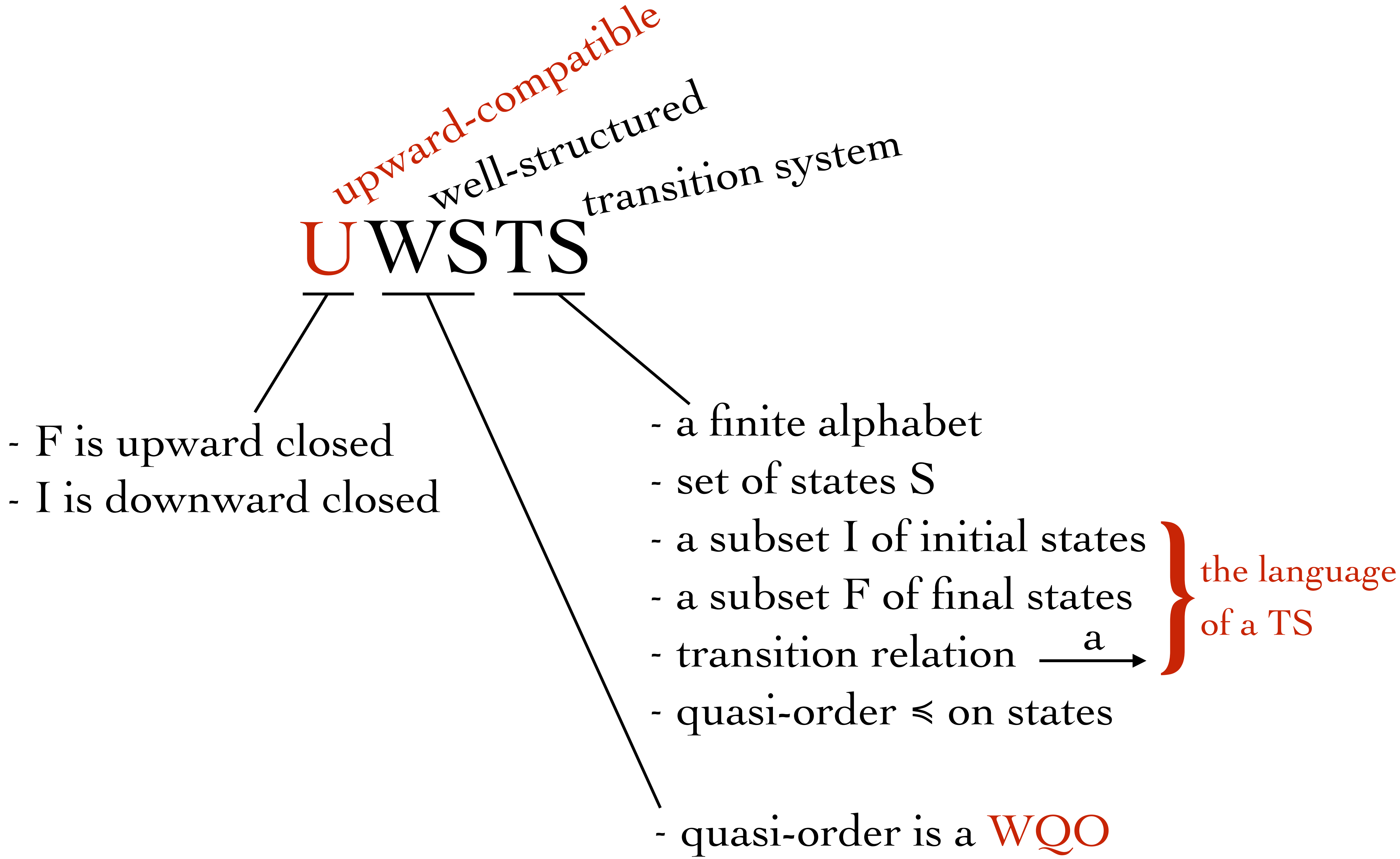
- F is upward closed

- a finite alphabet
- set of states S
- a subset I of initial states
- a subset F of final states
- transition relation $\xrightarrow{a}$
- quasi-order ≼ on states

the language of a TS

- quasi-order is a WQO

$^{U}/_{D}$WSTS: well-structured transition system



upward-compatible

well-structured

transition system

UWSTS

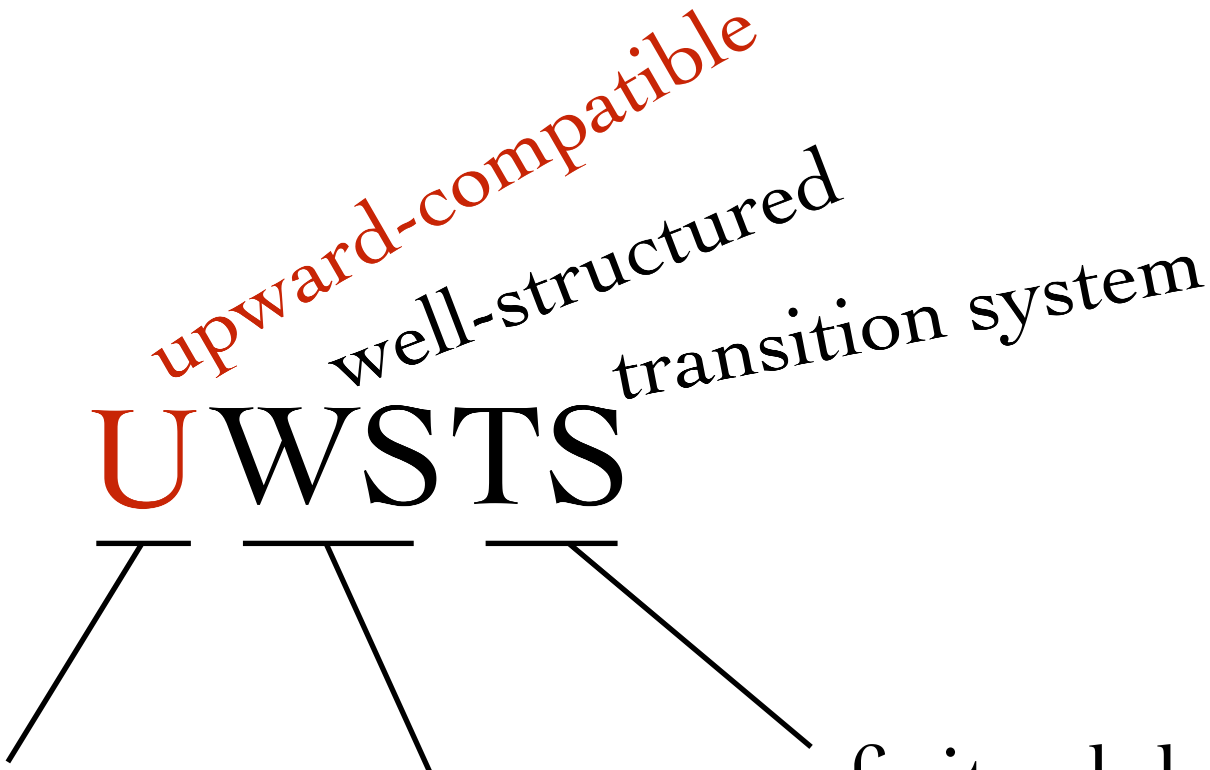
- F is upward closed
- I is downward closed

- a finite alphabet
- set of states S
- a subset I of initial states
- a subset F of final states
- transition relation $\xrightarrow{a}$
- quasi-order $\preccurlyeq$ on states

the language of a TS

- quasi-order is a WQO

$^{U}/_{D}$WSTS: well-structured transition system



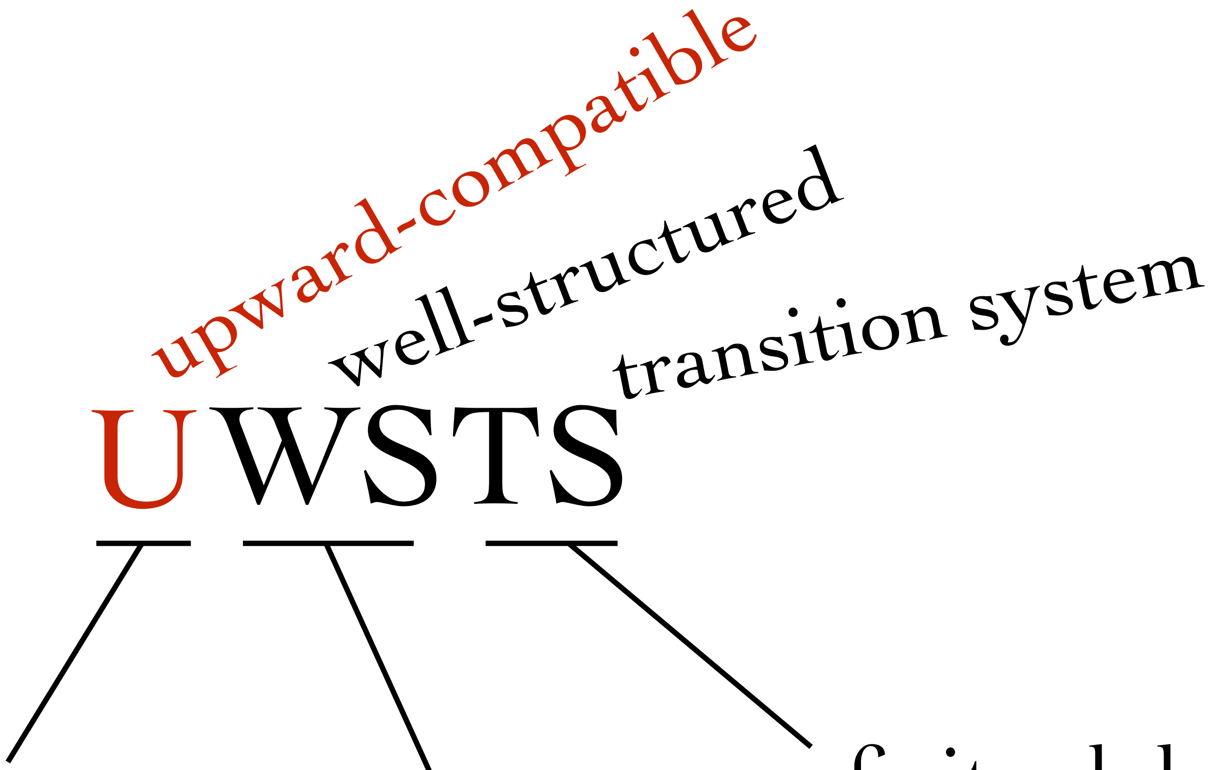
- F is upward closed
- I is downward closed
- upward-compatibility:

- a finite alphabet
- set of states S
- a subset I of initial states
- a subset F of final states
- transition relation $\xrightarrow{a}$
- quasi-order $\preccurlyeq$ on states

the language of a TS

- quasi-order is a WQO

$^{U}/_{D}$WSTS: well-structured transition system



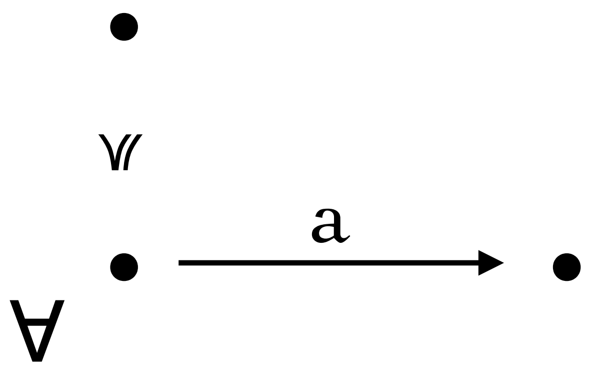
upward-compatible

well-structured

transition system

UWSTS

- F is upward closed
- I is downward closed
- upward-compatibility:

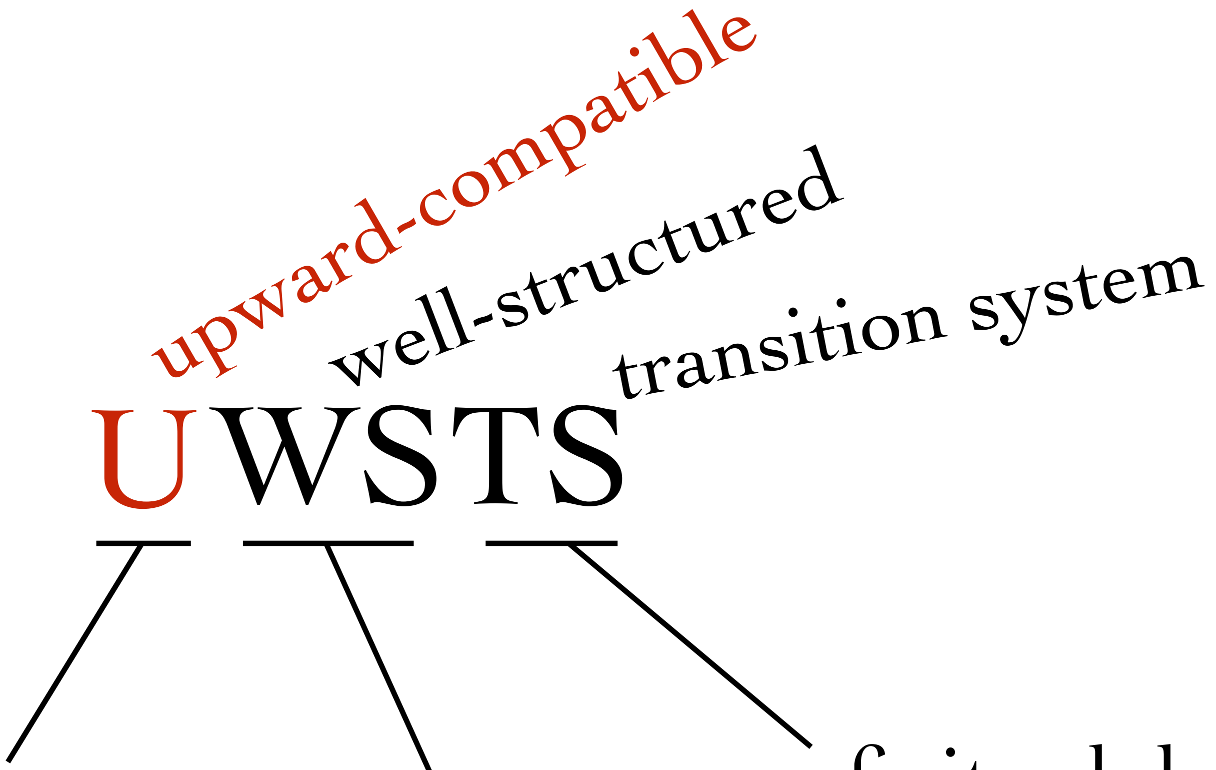


- a finite alphabet
- set of states S
- a subset I of initial states
- a subset F of final states
- transition relation $\xrightarrow{a}$
- quasi-order ≼ on states

} the language of a TS

- quasi-order is a WQO

$^{U}/_{D}$WSTS: well-structured transition system

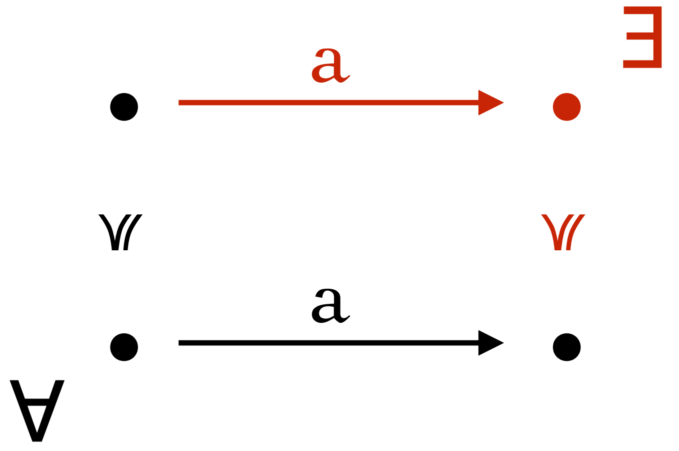

upward-compatible

well-structured

transition system

UWSTS

- F is upward closed
- I is downward closed
- upward-compatibility:

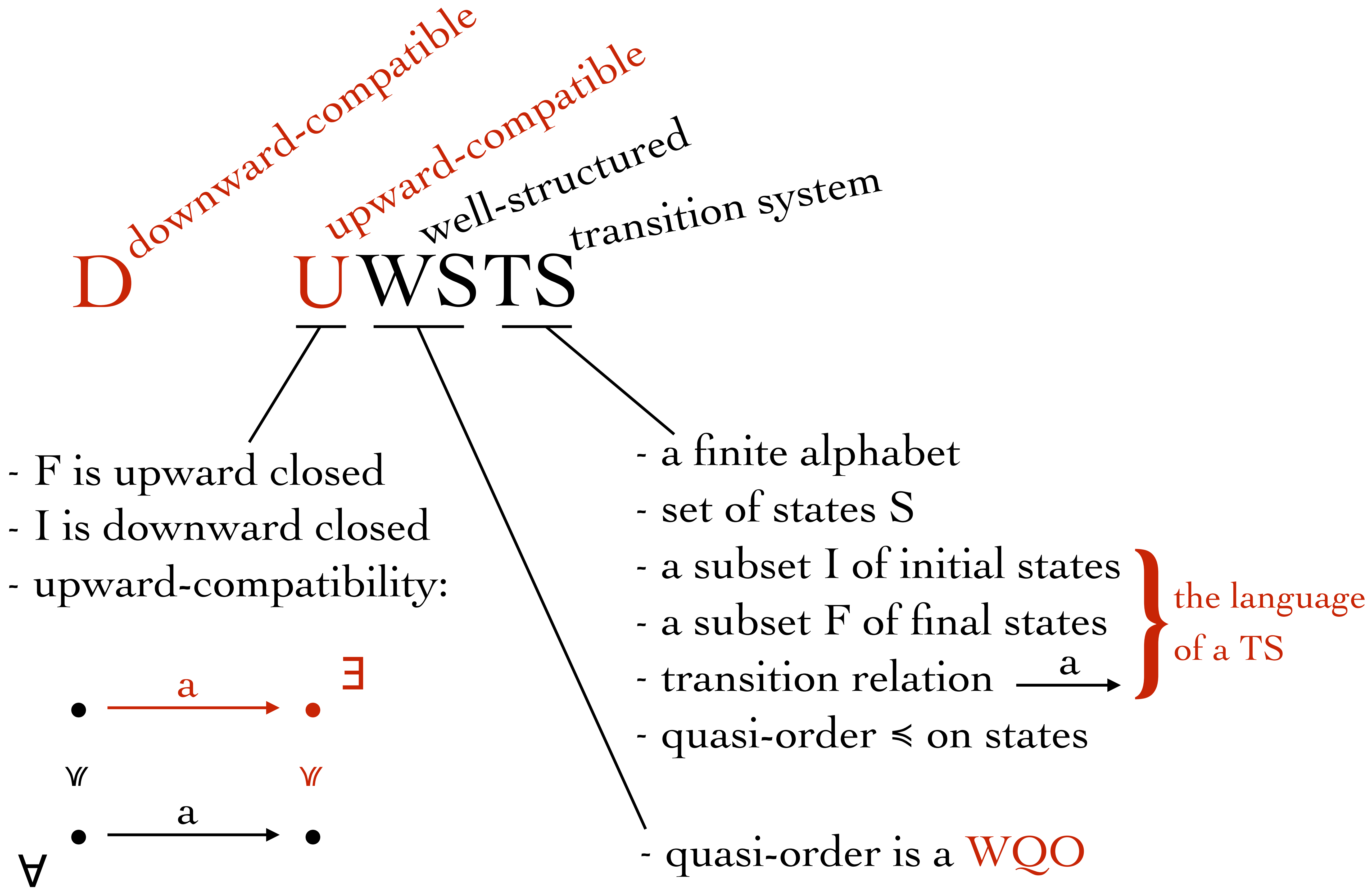


- a finite alphabet
- set of states S
- a subset I of initial states
- a subset F of final states
- transition relation $\xrightarrow{a}$
- quasi-order ≼ on states

the language of a TS

- quasi-order is a WQO

$^{U}/_{D}$WSTS: well-structured transition system



D downward-compatible

U upward-compatible

W well-structured

TS transition system

- F is upward closed
- I is downward closed
- upward-compatibility:

$$\begin{tikzcd} & \bullet \arrow[r, "a"] & \bullet & \exists \\ & \bullet \arrow[u, phantom, "\preccurlyeq"] \arrow[r, "a"] & \bullet \arrow[u, phantom, "\preccurlyeq"] & \\ \forall & & & \end{tikzcd}$$

- a finite alphabet
- set of states S
- a subset I of initial states
- a subset F of final states
- transition relation $\xrightarrow{a}$
- quasi-order ≼ on states

the language of a TS

- quasi-order is a WQO

WQO: well quasi order

Def: a quasi order is a **WQO** if it has:

- no infinite descending chain
- no infinite antichain

WQO: well quasi order

Def: a quasi order is a **WQO** if it has:

- no infinite descending chain
- no infinite antichain

Examples:

WQO: well quasi order

Def: a quasi order is a **WQO** if it has:

- no infinite descending chain
- no infinite antichain

Examples:

- Dickson: $\mathbb{N}^k$ ordered pointwise $(2, 3, 0) \leqslant (4, 3, 5)$

WQO: well quasi order

Def: a quasi order is a **WQO** if it has:

- no infinite descending chain
- no infinite antichain

Examples:

- Dickson: $\mathbb{N}^k$ ordered pointwise $(2, 3, 0) \preccurlyeq (4, 3, 5)$
- Higman: A^* ordered by word embedding age $\preccurlyeq$ prague

WQO: well quasi order

Def: a quasi order is a **WQO** if it has:

- no infinite descending chain
- no infinite antichain

Examples:

- Dickson: $\mathbb{N}^k$ ordered pointwise $(2, 3, 0) \preccurlyeq (4, 3, 5)$
- Higman: A^* ordered by word embedding age $\preccurlyeq$ prague
- Kruskal tree embedding

WQO: well quasi order

Def: a quasi order is a **WQO** if it has:

- no infinite descending chain
- no infinite antichain

Examples:

- Dickson: $\mathbb{N}^k$ ordered pointwise $(2, 3, 0) \preccurlyeq (4, 3, 5)$
- Higman: A^* ordered by word embedding age $\preccurlyeq$ prague
- Kruskal tree embedding
- Graph minor ordering

WQO: well quasi order

Def: a quasi order is a **WQO** if it has:

- no infinite descending chain
- no infinite antichain

Examples:

- Dickson: $\mathbb{N}^k$ ordered pointwise $(2, 3, 0) \leqslant (4, 3, 5)$
- Higman: A^* ordered by word embedding age $\leqslant$ prague
- Kruskal tree embedding
- Graph minor ordering

Def: a quasi order is an ω^2**-WQO** if its downward closed subsets (ordered by inclusion) are a WQO

UWSTS examples:

UWSTS examples:

- Petri nets, vector addition systems, and extensions thereof

UWSTS examples:

- Petri nets, vector addition systems, and extensions thereof
- lossy FIFO or counter automata

UWSTS examples:

- Petri nets, vector addition systems, and extensions thereof
- lossy FIFO or counter automata

- states = $\mathbb{N}^k$

UWSTS examples:

- Petri nets, vector addition systems, and extensions thereof
- lossy FIFO or counter automata

  - states = $\mathbb{N}^k$
  - I = initial vector$\downarrow$

UWSTS examples:

- Petri nets, vector addition systems, and extensions thereof
- lossy FIFO or counter automata

  - states = $\mathbb{N}^k$
  - I = initial vector$\downarrow$
  - F = final vector$\uparrow$

UWSTS examples:

- Petri nets, vector addition systems, and extensions thereof
- lossy FIFO or counter automata

- states = $\mathbb{N}^k$
- I = initial vector$\downarrow$
- F = final vector$\uparrow$
- transition relation by addition

UWSTS examples:

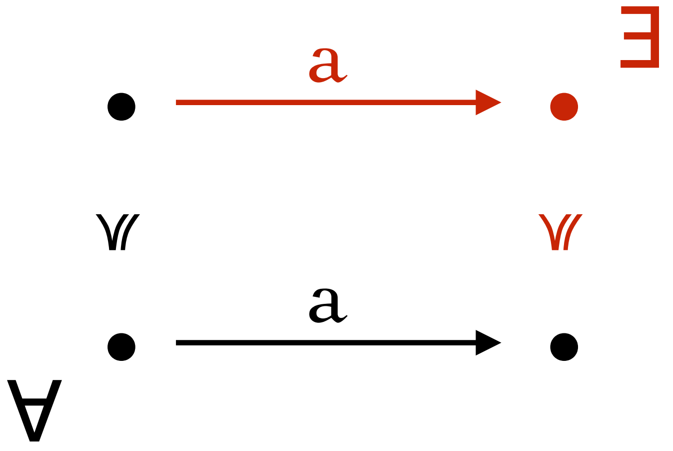
- Petri nets, vector addition systems, and extensions thereof
- lossy FIFO or counter automata

- states = $\mathbb{N}^k$
- I = initial vector$\downarrow$
- F = final vector$\uparrow$
- transition relation by addition
- Dickson order $\leqslant$

UWSTS examples:

- Petri nets, vector addition systems, and extensions thereof
- lossy FIFO or counter automata

upward compatibility:

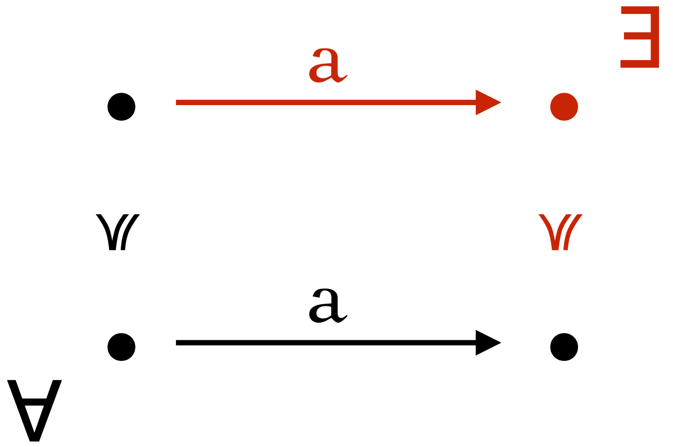


- states = $\mathbb{N}^k$
- I = initial vector↓
- F = final vector↑
- transition relation by addition
- Dickson order $\preccurlyeq$

UWSTS examples:

- Petri nets, vector addition systems, and extensions thereof
- lossy FIFO or counter automata

upward compatibility:

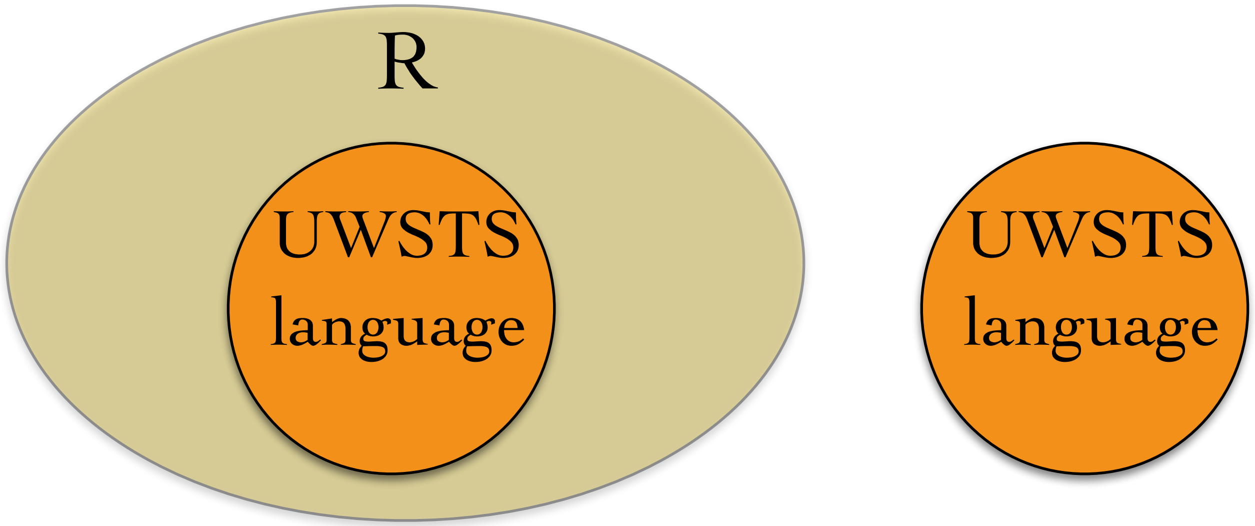


- states = $\mathbb{N}^k$
- I = initial vector↓
- F = final vector↑
- transition relation by addition
- Dickson order ≼

DWSTS examples:

- gainy FIFO or counter automata

Regular separability of ${}^{U}/_{D}$WSTS languages

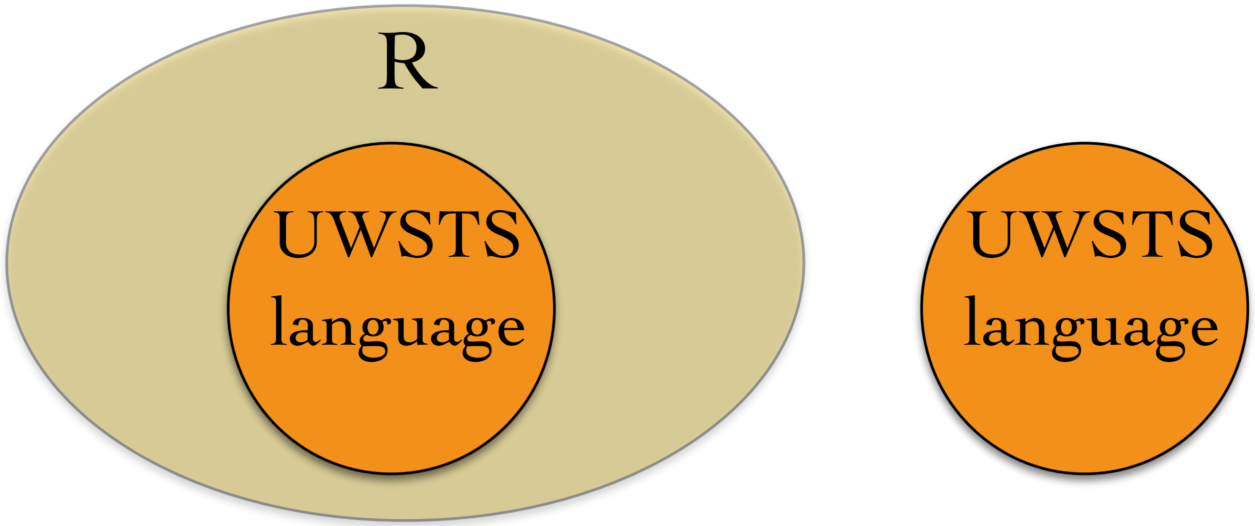


R

UWSTS language

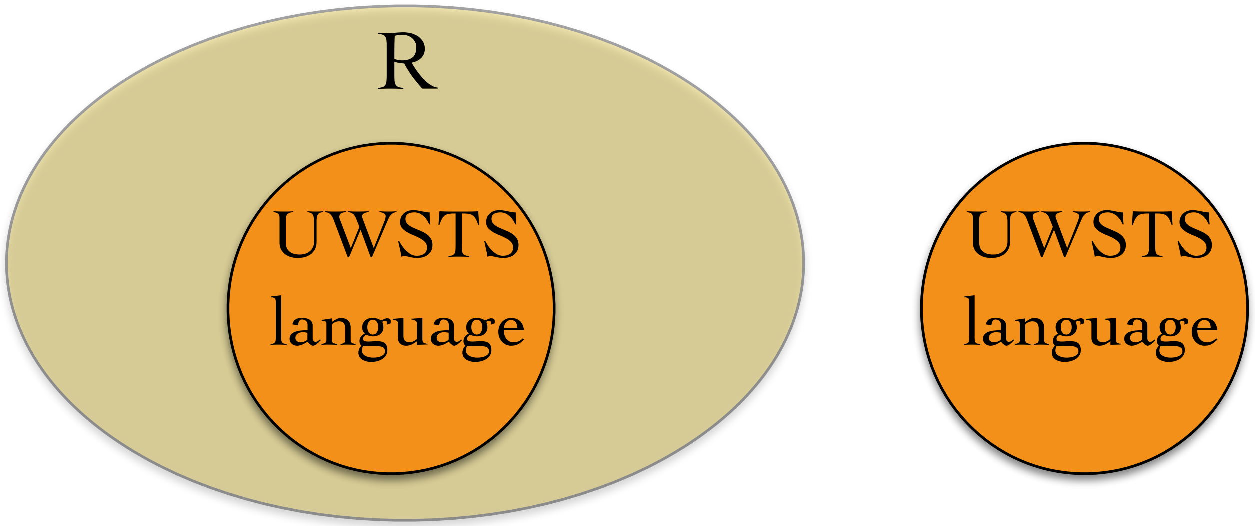
UWSTS language

Regular separability of $^{U}/_{D}$WSTS languages



Theorem: Every two disjoint UWSTS are regular-separable,
whenever one of them is **finitely-branching**.

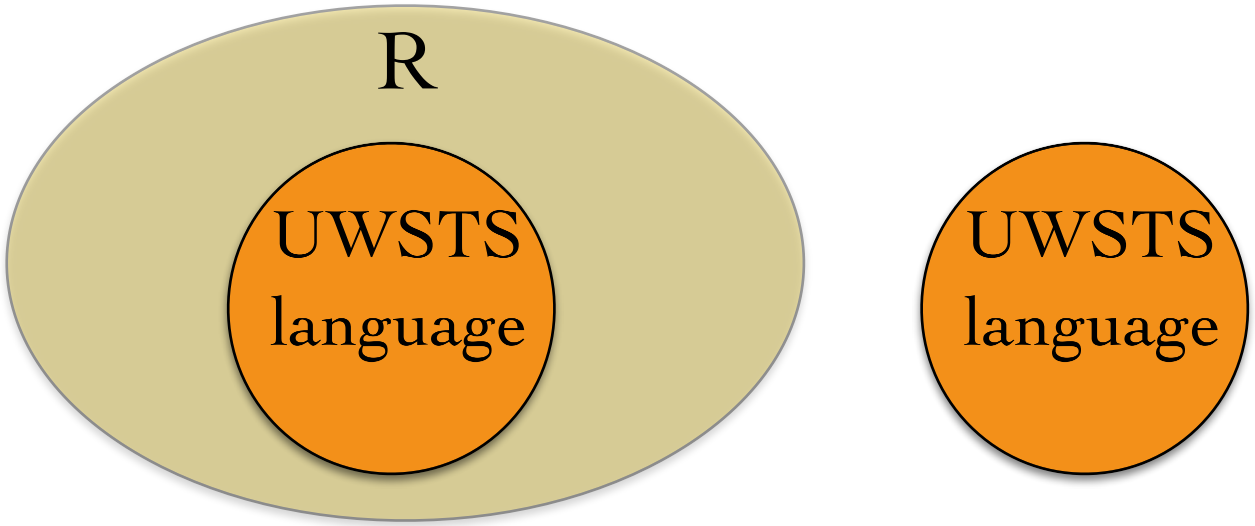
Regular separability of $^{U}/_{D}$WSTS languages



Theorem: Every two disjoint UWSTS are regular-separable, whenever one of them is **finitely-branching**.

Every two disjoint DWSTS are regular-separable, whenever one of them is **deterministic**.

Regular separability of $^{U}/_{D}$WSTS languages

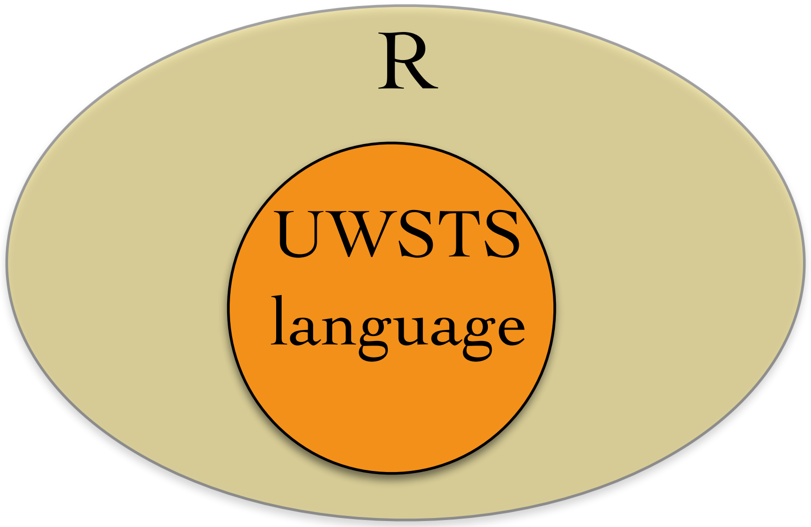


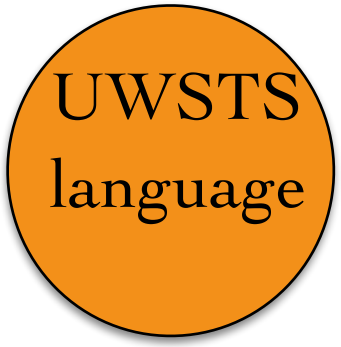
every state has finitely many a-successors

Theorem: Every two disjoint UWSTS are regular-separable, whenever one of them is **finitely-branching**.

Every two disjoint DWSTS are regular-separable, whenever one of them is **deterministic**.

Regular separability of $^{U}/_{D}$WSTS languages





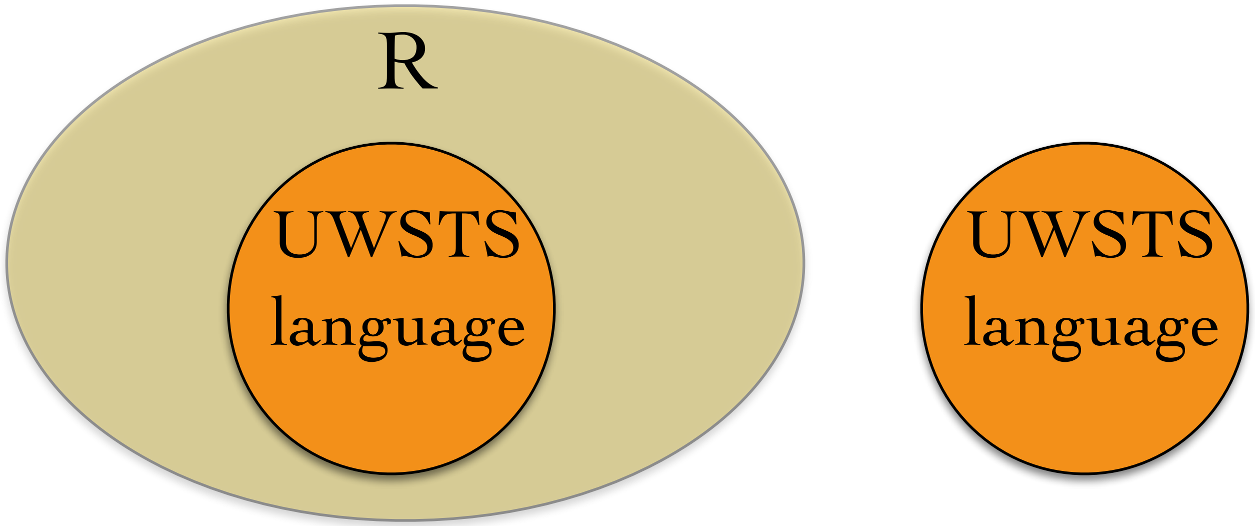
every state has
finitely many a-successors

Theorem: Every two disjoint UWSTS are regular-separable,
whenever one of them is **finitely-branching**.

Every two disjoint DWSTS are regular-separable,
whenever one of them is **deterministic**.

every state has
exactly one a-successor

Regular separability of $^{U}/_{D}$WSTS languages



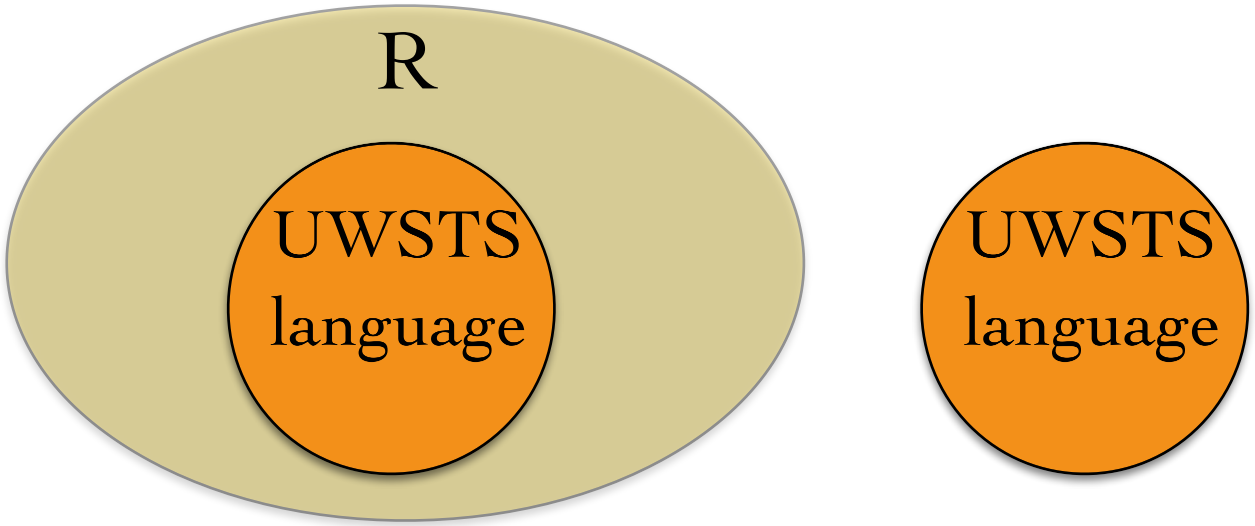
every state has
finitely many a-successors

Theorem: Every two disjoint UWSTS are regular-separable,
whenever one of them is **deterministic.**

Every two disjoint DWSTS are regular-separable,
whenever one of them is **deterministic**.

every state has
exactly one a-successor

Regular separability of $^{U}/_{D}$WSTS languages



every state has
finitely many a-successors

Theorem: Every two disjoint UWSTS are regular-separable,
whenever one of them is **deterministic.**

Every two disjoint DWSTS are regular-separable,
whenever one of them is **deterministic**.

every state has
exactly one a-successor

Corollary: Every two disjoint ω^2-UWSTS or ω^2-DWSTS languages are regular-separable.

Further consequences

Corollary: Every two disjoint ω^2-UWSTS or ω^2-DWSTS languages are regular-separable.

Further consequences

Corollary: Every two disjoint ω^2-UWSTS or ω^2-DWSTS languages are regular-separable.

Corollary: Every two disjoint languages of

Further consequences

Corollary: Every two disjoint ω^2-UWSTS or ω^2-DWSTS languages are regular-separable.

Corollary: Every two disjoint languages of

- plain/reset/transfer VASS (with coverability acceptance),

Further consequences

Corollary: Every two disjoint ω^2-UWSTS or ω^2-DWSTS languages are regular-separable.

Corollary: Every two disjoint languages of

- plain/reset/transfer VASS (with coverability acceptance),
- lossy FIFO/counter automata,

Further consequences

Corollary: Every two disjoint ω^2-UWSTS or ω^2-DWSTS languages are regular-separable.

Corollary: Every two disjoint languages of

- plain/reset/transfer VASS (with coverability acceptance),
- lossy FIFO/counter automata,
- …

Further consequences

Corollary: Every two disjoint ω^2-UWSTS or ω^2-DWSTS languages are regular-separable.

Corollary: Every two disjoint languages of

- plain/reset/transfer VASS (with coverability acceptance),
- lossy FIFO/counter automata,
- …

are regular-separable. Alike for gainy FIFO/counter automata.

Further consequences

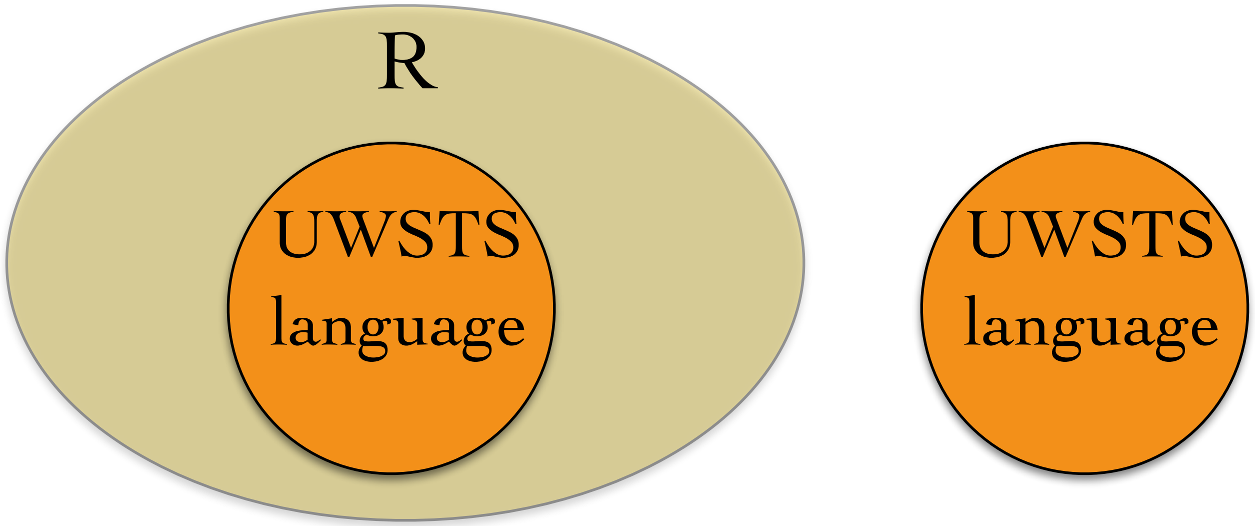
Corollary: Every two disjoint ω^2-UWSTS or ω^2-DWSTS languages are regular-separable.

Corollary: Every two disjoint languages of

- plain/reset/transfer VASS (with coverability acceptance),
- lossy FIFO/counter automata,
- …

are regular-separable. Alike for gainy FIFO/counter automata.

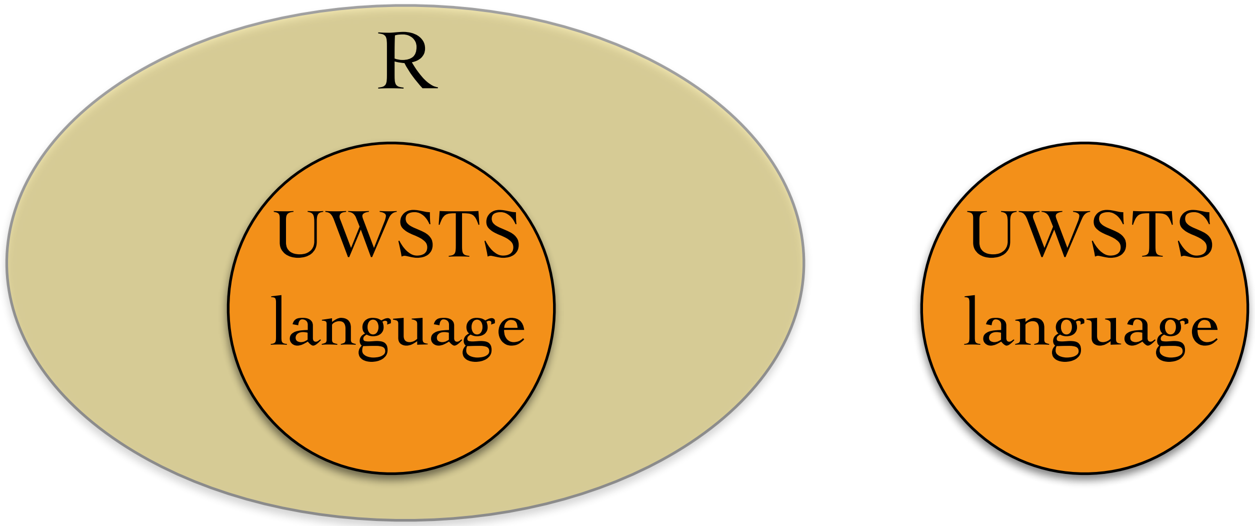
Corollary: No subclass of ${}^{U}/_{D}$WSTS languages closed under complement beyond regular languages.



Theorem: Every two disjoint UWSTS are regular-separable, whenever one of them is **deterministic**.

Every two disjoint DWSTS are regular-separable, whenever one of them is **deterministic**.

Proof: Main ingredients

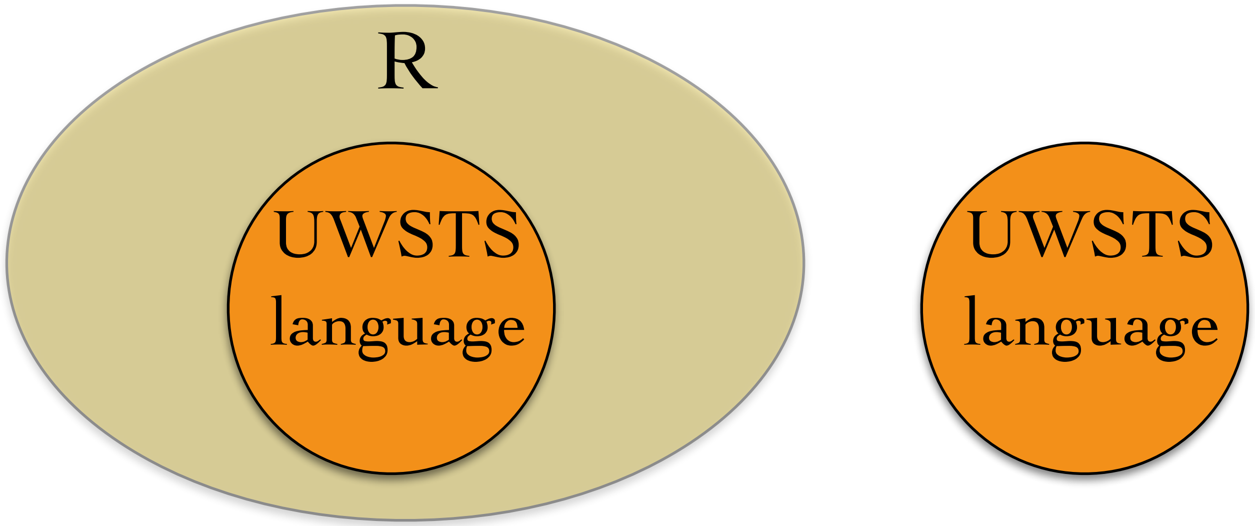


Theorem: Every two disjoint UWSTS are regular-separable, whenever one of them is **deterministic**.

Every two disjoint DWSTS are regular-separable, whenever one of them is **deterministic**.

Proof: Main ingredients

- **inductive invariant** in the synchronized product of $^{U}/_{D}$WSTS



Theorem: Every two disjoint UWSTS are regular-separable, whenever one of them is **deterministic**.

Every two disjoint DWSTS are regular-separable, whenever one of them is **deterministic**.

Proof: Main ingredients

- **inductive invariant** in the synchronized product of ${}^{U}/_{D}$WSTS
- **ideal completion** of a UWSTS

we could stop here…

Inductive invariant

Inductive invariant

Def: An **inductive invariant** in a UTS is a subset $X \subseteq S$ of states s.t.

Inductive invariant

Def: An **inductive invariant** in a UTS is a subset $X \subseteq S$ of states s.t.

- X is downward closed

Inductive invariant

Def: An **inductive invariant** in a UTS is a subset X ⊆ S of states s.t.

- X is downward closed
- I ⊆ X

Inductive invariant

Def: An **inductive invariant** in a UTS is a subset $X \subseteq S$ of states s.t.

- X is downward closed
- $I \subseteq X$
- $X \cap F = \varnothing$

Inductive invariant

Def: An **inductive invariant** in a UTS is a subset X ⊆ S of states s.t.

- X is downward closed
- I ⊆ X
- X ∩ F = ∅
- successors(X) ⊆ X

Inductive invariant

Def: An **inductive invariant** in a UTS is a subset $X \subseteq S$ of states s.t.

- X is downward closed
- $I \subseteq X$
- $X \cap F = \emptyset$
- successors(X) $\subseteq X$

Fact: Every empty-language UTS admits an inductive invariant, e.g.,

Inductive invariant

Def: An **inductive invariant** in a UTS is a subset X ⊆ S of states s.t.

- X is downward closed
- I ⊆ X
- X ∩ F = ∅
- successors(X) ⊆ X

Fact: Every empty-language UTS admits an inductive invariant, e.g.,

- the downward closure of the reachability set
- the complement of the backward reachability set

Inductive invariant

Def: An **inductive invariant** in a UTS is a subset $X \subseteq S$ of states s.t.

- X is downward closed
- $I \subseteq X$
- $X \cap F = \emptyset$
- successors$(X) \subseteq X$

Fact: Every empty-language UTS admits an inductive invariant, e.g.,

- the downward closure of the reachability set
- the complement of the backward reachability set

In particular, the **synchronized product** of two disjoint UTS admits one.

Inductive invariant

Def: An **inductive invariant** in a UTS is a subset $X \subseteq S$ of states s.t.

- X is downward closed
- $I \subseteq X$
- $X \cap F = \varnothing$
- successors(X) $\subseteq X$

Fact: Every empty-language UTS admits an inductive invariant, e.g.,

- the downward closure of the reachability set
- the complement of the backward reachability set

In particular, the **synchronized product** of two disjoint UTS admits one.

We will need **finitary** inductive invariants $Q{\downarrow}$, namely Q finite.

From inductive invariant to separator

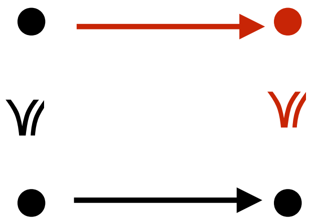
Key Lemma: If the synchronized product W×V of two UTS,
V deterministic, admits an inductive invariant Q↓, then
W and V are separated by an automaton with state space Q.

From inductive invariant to separator

Key Lemma: If the synchronized product W×V of two UTS, V deterministic, admits an inductive invariant Q↓, then W and V are separated by an automaton with state space Q.

$I \subseteq Q{\downarrow}$

Proof: We define automaton A to overapproximate W×V wrt ≼.

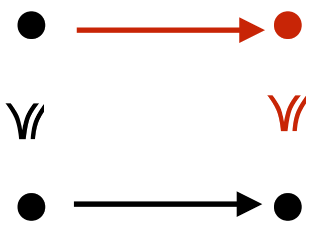


From inductive invariant to separator

Key Lemma: If the synchronized product W×V of two UTS,
V deterministic, admits an inductive invariant Q↓, then
W and V are separated by an automaton with state space Q.

$I \subseteq Q{\downarrow}$

Proof: We define automaton A to overapproximate W×V wrt ≼.
Final states of A: the W-component is final in W.



From inductive invariant to separator

Key Lemma: If the synchronized product W×V of two UTS,
V deterministic, admits an inductive invariant Q↓, then
W and V are separated by an automaton with state space Q.

I ⊆ Q↓

Proof: We define automaton A to overapproximate W×V wrt ≼.
Final states of A: the W-component is final in W.
Thus L(W) ⊆ L(A).

$$\begin{tikzcd}
\bullet \arrow[r, red] & \color{red}\bullet \\
\bullet \arrow[r] \arrow[u, phantom, sloped, "\preccurlyeq"] & \bullet \arrow[u, phantom, sloped, red, "\preccurlyeq"]
\end{tikzcd}$$

From inductive invariant to separator

Key Lemma: If the synchronized product W×V of two UTS,
V deterministic, admits an inductive invariant Q↓, then
W and V are separated by an automaton with state space Q.

I ⊆ Q↓

Proof: We define automaton A to overapproximate W×V wrt ≼.
Final states of A: the W-component is final in W.
Thus L(W) ⊆ L(A).

$$\begin{tikzcd}
\bullet \arrow[r] & \bullet \\
\bullet \arrow[u, phantom, "\rotatebox{90}{$\preccurlyeq$}"] \arrow[r] & \bullet \arrow[u, phantom, "\rotatebox{90}{$\preccurlyeq$}"]
\end{tikzcd}$$

Using determinacy of V,
the V-component of every state reached by A along some word

From inductive invariant to separator

Key Lemma: If the synchronized product W×V of two UTS,
V deterministic, admits an inductive invariant $Q{\downarrow}$, then
W and V are separated by an automaton with state space Q.

$I \subseteq Q{\downarrow}$

Proof: We define automaton A to overapproximate W×V wrt $\preccurlyeq$.
Final states of A: the W-component is final in W.
Thus $L(W) \subseteq L(A)$.

$$\begin{tikzcd}
\bullet \arrow[r] & \bullet \\
\bullet \arrow[r] \arrow[u, phantom, sloped, "\preccurlyeq"] & \bullet \arrow[u, phantom, sloped, "\preccurlyeq"]
\end{tikzcd}$$

Using determinacy of V,
the V-component of every state reached by A along some word
$\preccurlyeq$-dominates the unique state reached by V along this word.

From inductive invariant to separator

Key Lemma: If the synchronized product W×V of two UTS,
V deterministic, admits an inductive invariant Q↓, then
W and V are separated by an automaton with state space Q.

I ⊆ Q↓

Proof: We define automaton A to overapproximate W×V wrt ≼.
Final states of A: the W-component is final in W.
Thus L(W) ⊆ L(A).

$$\begin{tikzcd}
\bullet \arrow[r] & \bullet \\
\bullet \arrow[r] \arrow[u, phantom, sloped, "\preccurlyeq"] & \bullet \arrow[u, phantom, sloped, "\preccurlyeq"]
\end{tikzcd}$$

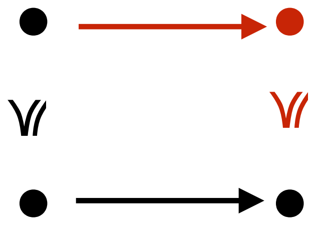
Using determinacy of V,
the V-component of every state reached by A along some word
≼-dominates the unique state reached by V along this word.
Thus L(A) ∩ L(V) = ∅. □

From inductive invariant to separator

Key Lemma: If the synchronized product W×V of two UTS,
V deterministic, admits an inductive invariant $Q\downarrow$, then
W and V are separated by an automaton with state space Q.

$I \subseteq Q\downarrow$

Proof: We define automaton A to overapproximate W×V wrt $\preccurlyeq$.
Final states of A: the W-component is final in W.
Thus $L(W) \subseteq L(A)$.



Using determinacy of V,
the V-component of every state reached by A along some word
$\preccurlyeq$-dominates the unique state reached by V along this word.
Thus $L(A) \cap L(V) = \emptyset$. □

It remains to demonstrate existence of a finite Q.

Regular separability of DWSTS languages

Key Lemma: If the synchronized product W×V of two UTS,
V deterministic, admits an inductive invariant Q↓, then
W and V are separated by an automaton with state space Q.

$$\Downarrow$$

Theorem: Every two disjoint DWSTS are regular-separable,
whenever one of them is **deterministic**.

Regular separability of DWSTS languages

Key Lemma: If the synchronized product W×V of two UTS, V deterministic, admits an inductive invariant Q↓, then W and V are separated by an automaton with state space Q.

⇓

Theorem: Every two disjoint DWSTS are regular-separable, whenever one of them is **deterministic**.

Proof: Apply Key Lemma to **inverses** of DWSTS which are UTS.

Regular separability of DWSTS languages

Key Lemma: If the synchronized product W×V of two UTS, V deterministic, admits an inductive invariant Q↓, then W and V are separated by an automaton with state space Q.

$$\Downarrow$$

Theorem: Every two disjoint DWSTS are regular-separable, whenever one of them is **deterministic**.

Proof: Apply Key Lemma to **inverses** of DWSTS which are UTS. Finite min of upward closed set inverses to finite max of downward closed sets. □

Ideal completion of a UWSTS

Recall: We need a **finitary** inductive invariant $Q\downarrow$, for Q finite.

Ideal completion of a UWSTS

Recall: We need a **finitary** inductive invariant $Q{\downarrow}$, for Q finite.

Def: An **ideal** in a quasi-order is any downward closed directed subset thereof. $(3, \omega, 4)$

Ideal completion of a UWSTS

Recall: We need a **finitary** inductive invariant $Q\downarrow$, for Q finite.

Def: An **ideal** in a quasi-order is any downward closed directed subset thereof. $(3, \omega, 4)$

Finite ideal decomposition: Every downward closed subset of a WQO is a finite union of ideals.

Ideal completion of a UWSTS

Recall: We need a **finitary** inductive invariant $Q{\downarrow}$, for Q finite.

Def: An **ideal** in a quasi-order is any downward closed directed subset thereof. $(3, \omega, 4)$

Finite ideal decomposition: Every downward closed subset of a WQO is a finite union of ideals.

Ideal completion: extend quasi-order by all its ideals.

Ideal completion of a UWSTS

Recall: We need a **finitary** inductive invariant Q↓, for Q finite.

Def: An **ideal** in a quasi-order is any downward closed directed subset thereof. $(3, \omega, 4)$

Finite ideal decomposition: Every downward closed subset of a WQO is a finite union of ideals.

Ideal completion: extend quasi-order by all its ideals.

Fact 1: Ideal completion of a (deterministic) UWSTS is a language-equivalent (deterministic) UTS.

Ideal completion of a UWSTS

Recall: We need a **finitary** inductive invariant Q↓, for Q finite.

Def: An **ideal** in a quasi-order is any downward closed directed subset thereof. $(3, \omega, 4)$

Finite ideal decomposition: Every downward closed subset of a WQO is a finite union of ideals.

Ideal completion: extend quasi-order by all its ideals.

Fact 1: Ideal completion of a (deterministic) UWSTS is a language-equivalent (deterministic) UTS.

Fact 2: Ideal completion commutes with synchronized product.

Ideal completion of a UWSTS

Recall: We need a **finitary** inductive invariant Q↓, for Q finite.

Def: An **ideal** in a quasi-order is any downward closed directed subset thereof. $(3, \omega, 4)$

Finite ideal decomposition: Every downward closed subset of a WQO is a finite union of ideals.

Ideal completion: extend quasi-order by all its ideals.

Fact 1: Ideal completion of a (deterministic) UWSTS is a language-equivalent (deterministic) UTS.

Fact 2: Ideal completion commutes with synchronized product.

Fact 3: For every inductive invariant in a UWSTS, its finite ideal decomposition is a **finitary** inductive invariant in the ideal completion of this UWSTS.

Regular separability of UWSTS languages

Key Lemma: If the synchronized product W×V of two UTS,
V deterministic, admits an inductive invariant Q↓, then
W and V are separated by an automaton with state space Q.

$$\Downarrow$$

Theorem: Every two disjoint UWSTS are regular-separable,
whenever one of them is **deterministic**.

Regular separability of UWSTS languages

Key Lemma: If the synchronized product $W \times V$ of two UTS,
V deterministic, admits an inductive invariant $Q{\downarrow}$, then
W and V are separated by an automaton with state space Q.

$$\Downarrow$$

Theorem: Every two disjoint UWSTS are regular-separable,
whenever one of them is **deterministic**.

Fact 1: Ideal completion of a (deterministic) UWSTS is
a language-equivalent (deterministic) UTS.

Proof: Apply Key Lemma to the ideal completions of the UWSTS.

Regular separability of UWSTS languages

Key Lemma: If the synchronized product W×V of two UTS,
V deterministic, admits an inductive invariant Q↓, then
W and V are separated by an automaton with state space Q.

⇓

Theorem: Every two disjoint UWSTS are regular-separable,
whenever one of them is **deterministic**.

Fact 1: Ideal completion of a (deterministic) UWSTS is a language-equivalent (deterministic) UTS.

Proof: Apply Key Lemma to the ideal completions of the UWSTS.
Synchronized product of idea completions, isomorphic to

Regular separability of UWSTS languages

Key Lemma: If the synchronized product W×V of two UTS,
V deterministic, admits an inductive invariant Q↓, then
W and V are separated by an automaton with state space Q.

⇓

Theorem: Every two disjoint UWSTS are regular-separable,
whenever one of them is **deterministic**.

Fact 1: Ideal completion of a (deterministic) UWSTS is
a language-equivalent (deterministic) UTS.

Proof: Apply Key Lemma to the ideal completions of the UWSTS.
Synchronized product of idea completions, isomorphic to
ideal completion of synchronized product,

Fact 2: Ideal completion commutes
with synchronized product.

Regular separability of UWSTS languages

Key Lemma: If the synchronized product W×V of two UTS,
V deterministic, admits an inductive invariant $Q{\downarrow}$, then
W and V are separated by an automaton with state space Q.

$$\Downarrow$$

Theorem: Every two disjoint UWSTS are regular-separable,
whenever one of them is **deterministic**.

Fact 1: Ideal completion of a (deterministic) UWSTS is
a language-equivalent (deterministic) UTS.

Proof: Apply Key Lemma to the ideal completions of the UWSTS.
Synchronized product of idea completions, isomorphic to
ideal completion of synchronized product,
admits a finitary inductive invariant.

Fact 2: Ideal completion commutes
with synchronized product. □

Fact 3: For every inductive invariant in a UWSTS,
its finite ideal decomposition is a **finitary** inductive invariant
in the ideal completion of this UWSTS.

Language expressibility $^{U}/_{D}$WSTS

Language expressibility $^{U}/_{D}$WSTS

Theorem: The following relations between the language classes:

Language expressibility $^{U}/_{D}$WSTS

Theorem: The following relations between the language classes:

- ω^2-UWSTS $\subseteq$ det. UWSTS $=$ fin-bran. UWSTS $\subseteq$ all UWSTS
- ω^2-DWSTS $\subseteq$ det. DWSTS $\subseteq$ fin-bran. DWSTS $=$ all DWSTS

Language expressibility $^{U}\!/_{D}$WSTS

Theorem: The following relations between the language classes:

- ω^2-UWSTS $\subseteq$ det. UWSTS $=$ fin-bran. UWSTS $\subseteq$ all UWSTS
- ω^2-DWSTS $\subseteq$ det. DWSTS $\subseteq$ fin-bran. DWSTS $=$ all DWSTS
- ω^2-UWSTS $\subseteq_{\text{rev}}$ det. DWSTS
- ω^2-DWSTS $\subseteq_{\text{rev}}$ det. UWSTS

Left for future

Left for future

Theorem: Every two disjoint UWSTS are regular-separable,
whenever one of them is **finitely-branching**.

Every two disjoint DWSTS are regular-separable,
whenever one of them is **deterministic**.

Left for future

Theorem: Every two disjoint UWSTS are regular-separable,
whenever one of them is **finitely-branching**.

Every two disjoint DWSTS are regular-separabl
whenever one of them is **determini**

Can these assumptions be dropped?

Left for future

Theorem: Every two disjoint UWSTS are regular-separable,
whenever one of them is **finitely-branching**.

Every two disjoint DWSTS are regular-separabl
whenever one of them is **determini**

Can these assumptions be dropped?

Theorem: The following relations between the language classes:

- ω^2-UWSTS $\subseteq$ det. UWSTS $=$ fin-bran. UWSTS $\subseteq$ all UWSTS
- ω^2-DWSTS $\subseteq$ det. DWSTS $\subseteq$ fin-bran. DWSTS $=$ all DWSTS
- ω^2-UWSTS $\subseteq_{\text{rev}}$ det. DWSTS
- ω^2-DWSTS $\subseteq_{\text{rev}}$ det. UWSTS

Left for future

Theorem: Every two disjoint UWSTS are regular-separable,
whenever one of them is **finitely-branching**.

Every two disjoint DWSTS are regular-separabl
whenever one of them is **determini**

Can these assumptions be dropped?

Theorem: The following relations between the language classes:

- ω^2-UWSTS $\subseteq$ det. UWSTS $=$ fin-bran. UWSTS $\subseteq$ all UWSTS
- ω^2-DWSTS $\subseteq$ det. DWSTS $\subseteq$ fin-bran. DWSTS $=$ all DWSTS
- ω^2-UWSTS $\subseteq_{\text{rev}}$ det. DWSTS
- ω^2-DWSTS $\subseteq_{\text{rev}}$ det. UWSTS

Are the inclusions strict?

Left for future

Theorem: Every two disjoint UWSTS are regular-separable,
whenever one of them is **finitely-branching**.

Every two disjoint DWSTS are regular-separabl
whenever one of them is **determini**

Can these assumptions be dropped?

Theorem: The following relations between the language classes:

- ω^2-UWSTS $\subseteq$ det. UWSTS $=$ fin-bran. UWSTS $\subseteq$ all UWSTS
- ω^2-DWSTS $\subseteq$ det. DWSTS $\subseteq$ fin-bran. DWSTS $=$ all DWSTS
- ω^2-UWSTS $\subseteq_{\text{rev}}$ det. DWSTS
- ω^2-DWSTS $\subseteq_{\text{rev}}$ det. UWSTS

Are the inclusions strict?

Obvious generalizations:

Left for future

Theorem: Every two disjoint UWSTS are regular-separable,
whenever one of them is **finitely-branching**.

Every two disjoint DWSTS are regular-separabl
whenever one of them is **determini**

Can these assumptions be dropped?

Theorem: The following relations between the language classes:

- ω^2-UWSTS $\subseteq$ det. UWSTS $=$ fin-bran. UWSTS $\subseteq$ all UWSTS
- ω^2-DWSTS $\subseteq$ det. DWSTS $\subseteq$ fin-bran. DWSTS $=$ all DWSTS
- ω^2-UWSTS $\subseteq_{\text{rev}}$ det. DWSTS
- ω^2-DWSTS $\subseteq_{\text{rev}}$ det. UWSTS

Are the inclusions strict?

Obvious generalizations:

- languages of trees instead of words (BVASS coverability languages)

Left for future

Theorem: Every two disjoint UWSTS are regular-separable,
whenever one of them is **finitely-branching**.

Every two disjoint DWSTS are regular-separable,
whenever one of them is **deterministic**.

Can these assumptions be dropped?

Theorem: The following relations between the language classes:

- ω^2-UWSTS $\subseteq$ det. UWSTS = fin-bran. UWSTS $\subseteq$ all UWSTS
- ω^2-DWSTS $\subseteq$ det. DWSTS $\subseteq$ fin-bran. DWSTS = all DWSTS
- ω^2-UWSTS $\subseteq_{\text{rev}}$ det. DWSTS
- ω^2-DWSTS $\subseteq_{\text{rev}}$ det. UWSTS

Are the inclusions strict?

Obvious generalizations:

- languages of trees instead of words (BVASS coverability languages)
- orbit-finite alphabets instead of finite ones (data VAS)

Left for future

Theorem: Every two disjoint UWSTS are regular-separable,
whenever one of them is **finitely-branching**.

Every two disjoint DWSTS are regular-separabl
whenever one of them is **determini**

Can these assumptions be dropped?

Theorem: The following relations between the language classes:

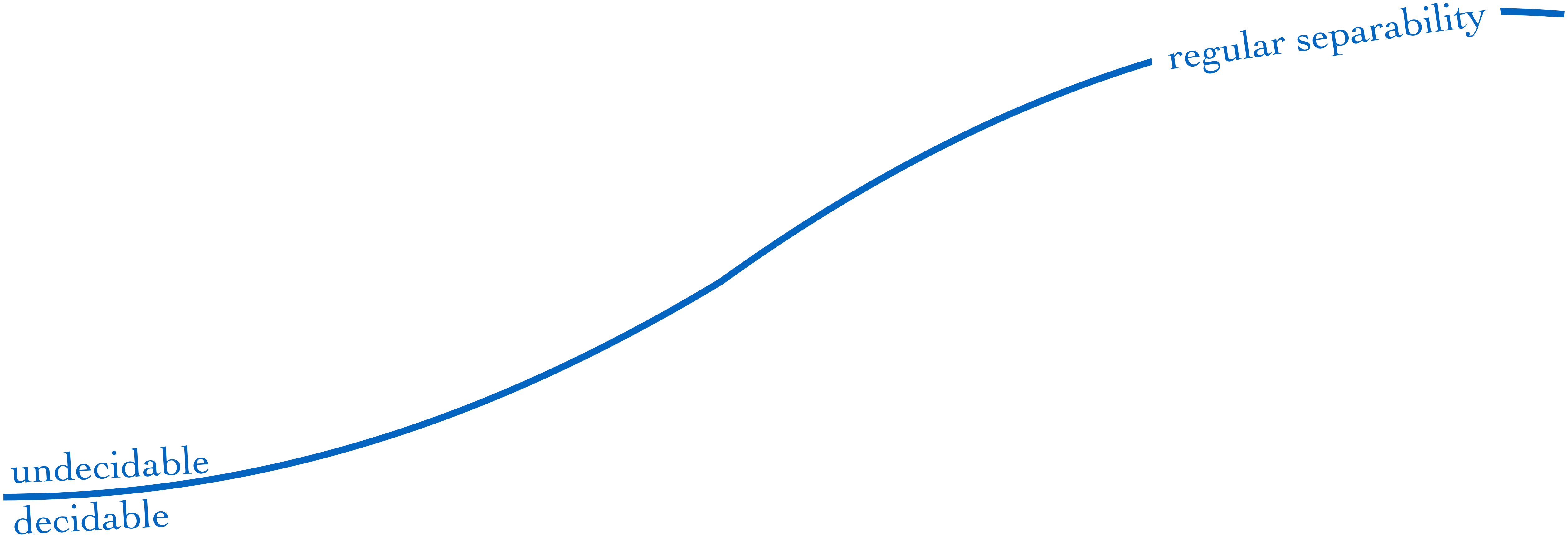
- ω^2-UWSTS $\subseteq$ det. UWSTS = fin-bran. UWSTS $\subseteq$ all UWSTS
- ω^2-DWSTS $\subseteq$ det. DWSTS $\subseteq$ fin-bran. DWSTS = all DWSTS
- ω^2-UWSTS $\subseteq_{\mathrm{rev}}$ det. DWSTS
- ω^2-DWSTS $\subseteq_{\mathrm{rev}}$ det. UWSTS

Are the inclusions strict?

Obvious generalizations:

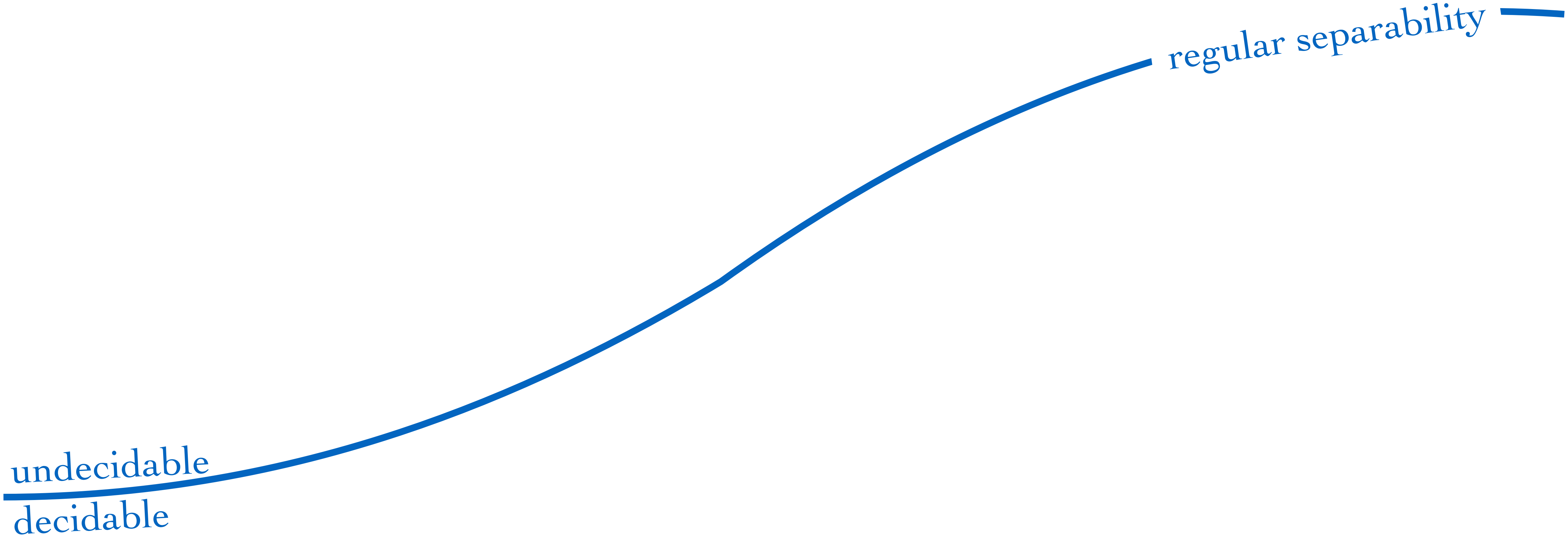
- languages of trees instead of words (BVASS coverability languages)
- orbit-finite alphabets instead of finite ones (data VAS)
- …

Regular separability as a decision problem



Regular separability as a decision problem

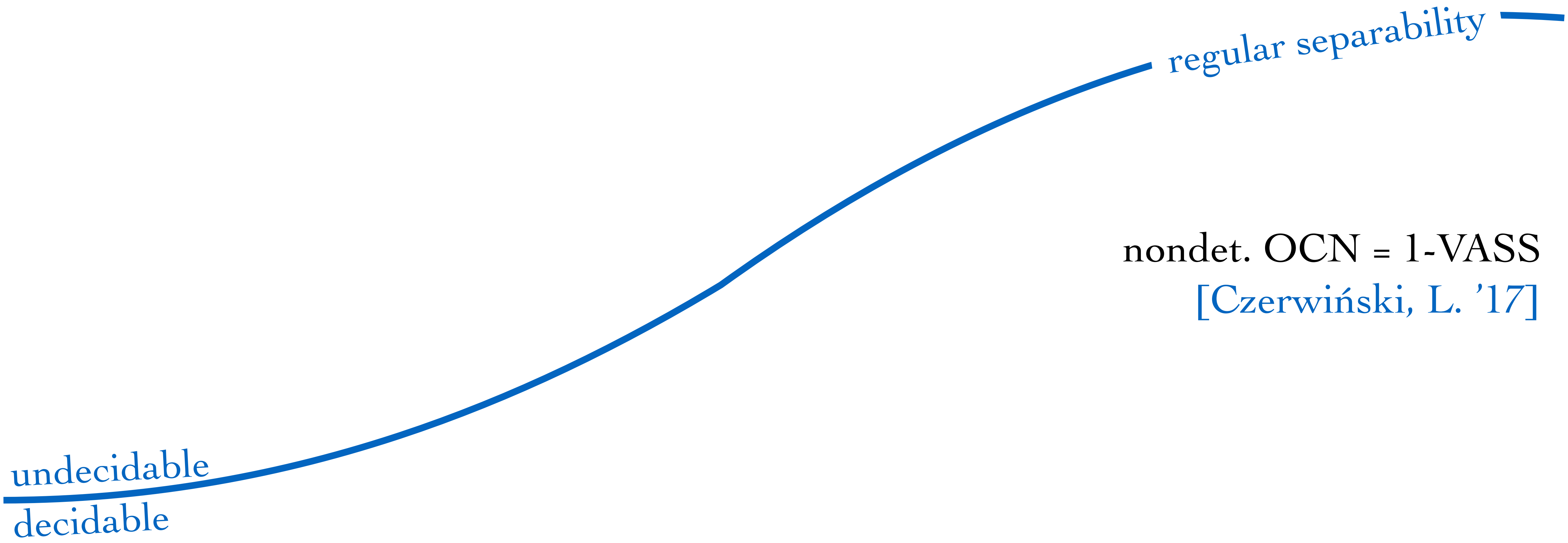
nondet. PDA [Szymanski, Williams '76]
[Hunt '82]



Regular separability as a decision problem

nondet. PDA [Szymanski, Williams '76]
[Hunt '82]

nondet. OCA [Czerwiński, L. '17]



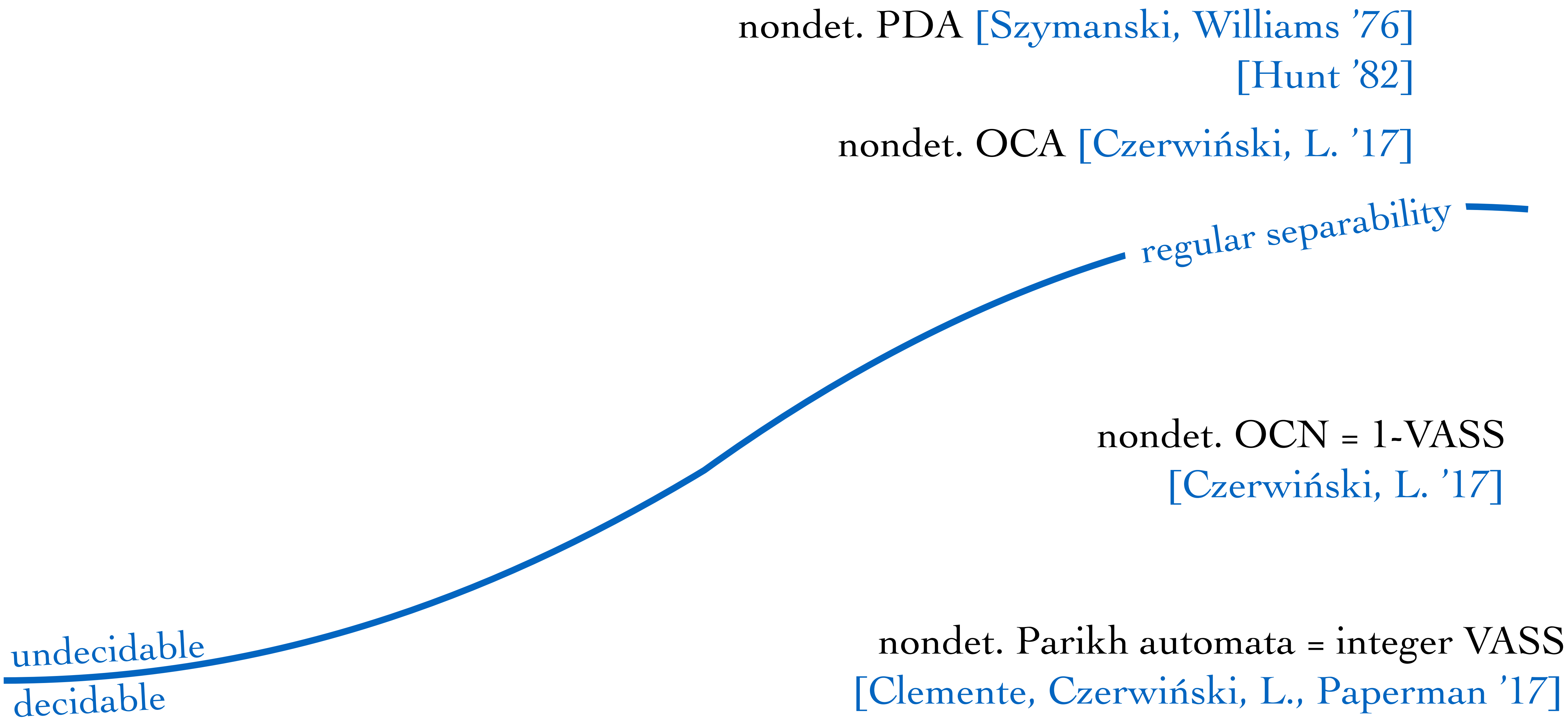
regular separability

undecidable

decidable

nondet. OCN = 1-VASS
[Czerwiński, L. '17]

Regular separability as a decision problem



nondet. PDA [Szymanski, Williams '76]
[Hunt '82]

nondet. OCA [Czerwiński, L. '17]

regular separability

nondet. OCN = 1-VASS
[Czerwiński, L. '17]

undecidable

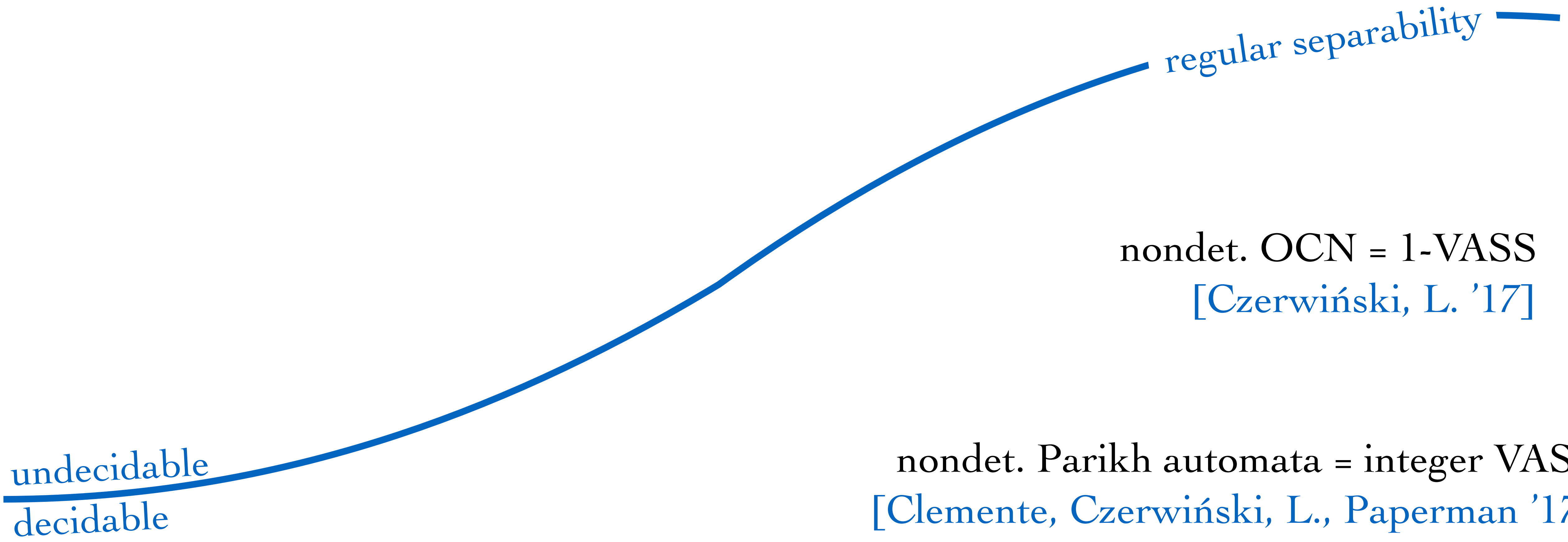
decidable

nondet. Parikh automata = integer VASS
[Clemente, Czerwiński, L., Paperman '17]

Regular separability as a decision problem

nondet. PDA [Szymanski, Williams '76]
[Hunt '82]

nondet. OCA [Czerwiński, L. '17]



nondet. OCN = 1-VASS
[Czerwiński, L. '17]

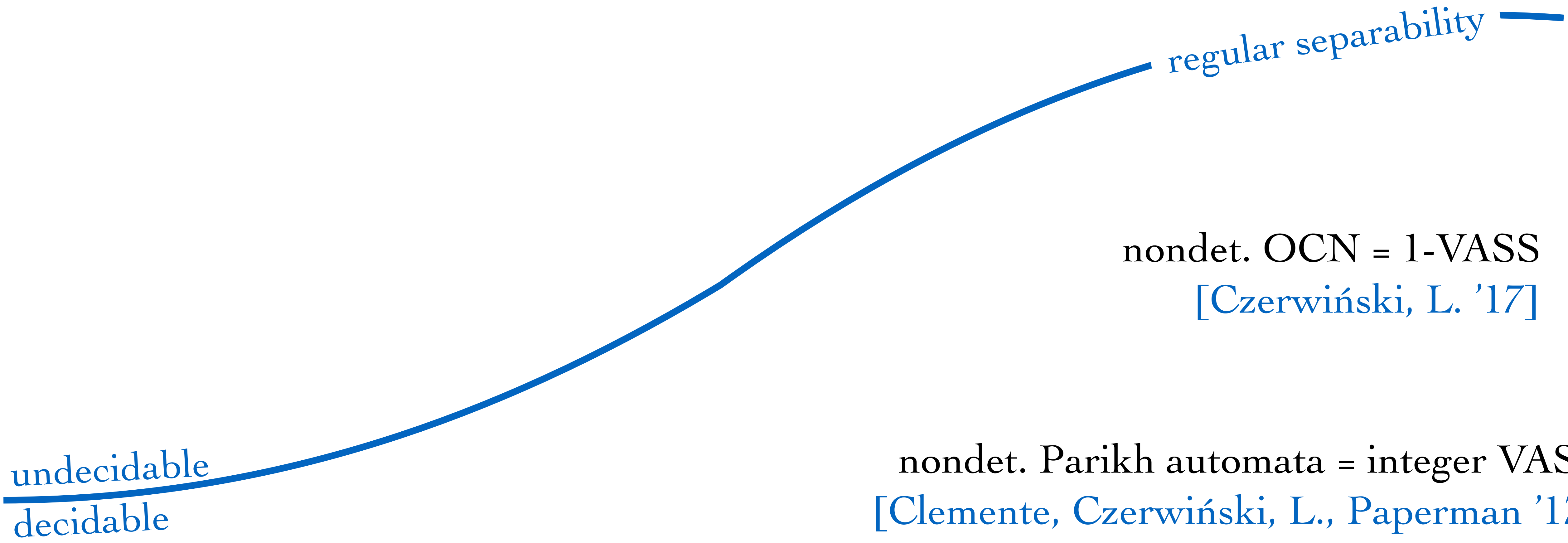
nondet. Parikh automata = integer VASS
[Clemente, Czerwiński, L., Paperman '17]

commutative closures of VASS
[Clemente, Czerwiński, L., Paperman '17]

Regular separability as a decision problem

nondet. PDA [Szymanski, Williams ’76]
[Hunt ’82]

nondet. OCA [Czerwiński, L. ’17]



nondet. OCN = 1-VASS
[Czerwiński, L. ’17]

nondet. Parikh automata = integer VASS
[Clemente, Czerwiński, L., Paperman ’17]

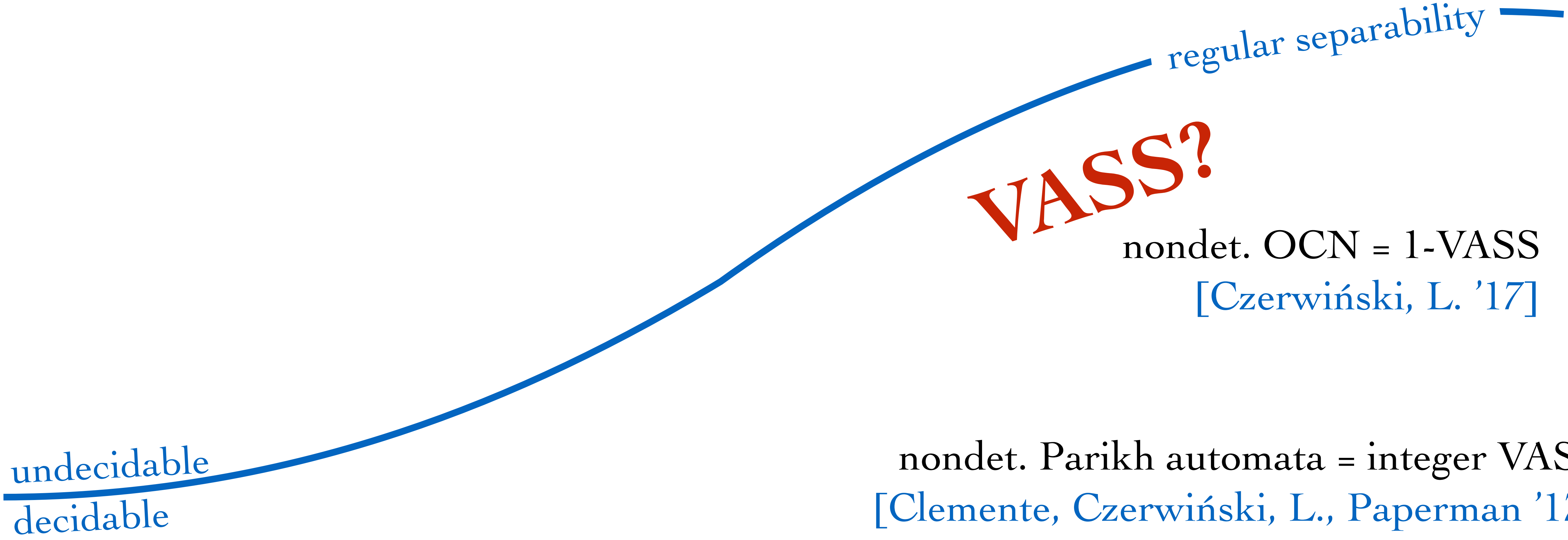
commutative closures of VASS
[Clemente, Czerwiński, L., Paperman ’17]

coverability VASS

Regular separability as a decision problem

nondet. PDA [Szymanski, Williams ’76]
[Hunt ’82]

nondet. OCA [Czerwiński, L. ’17]



VASS?

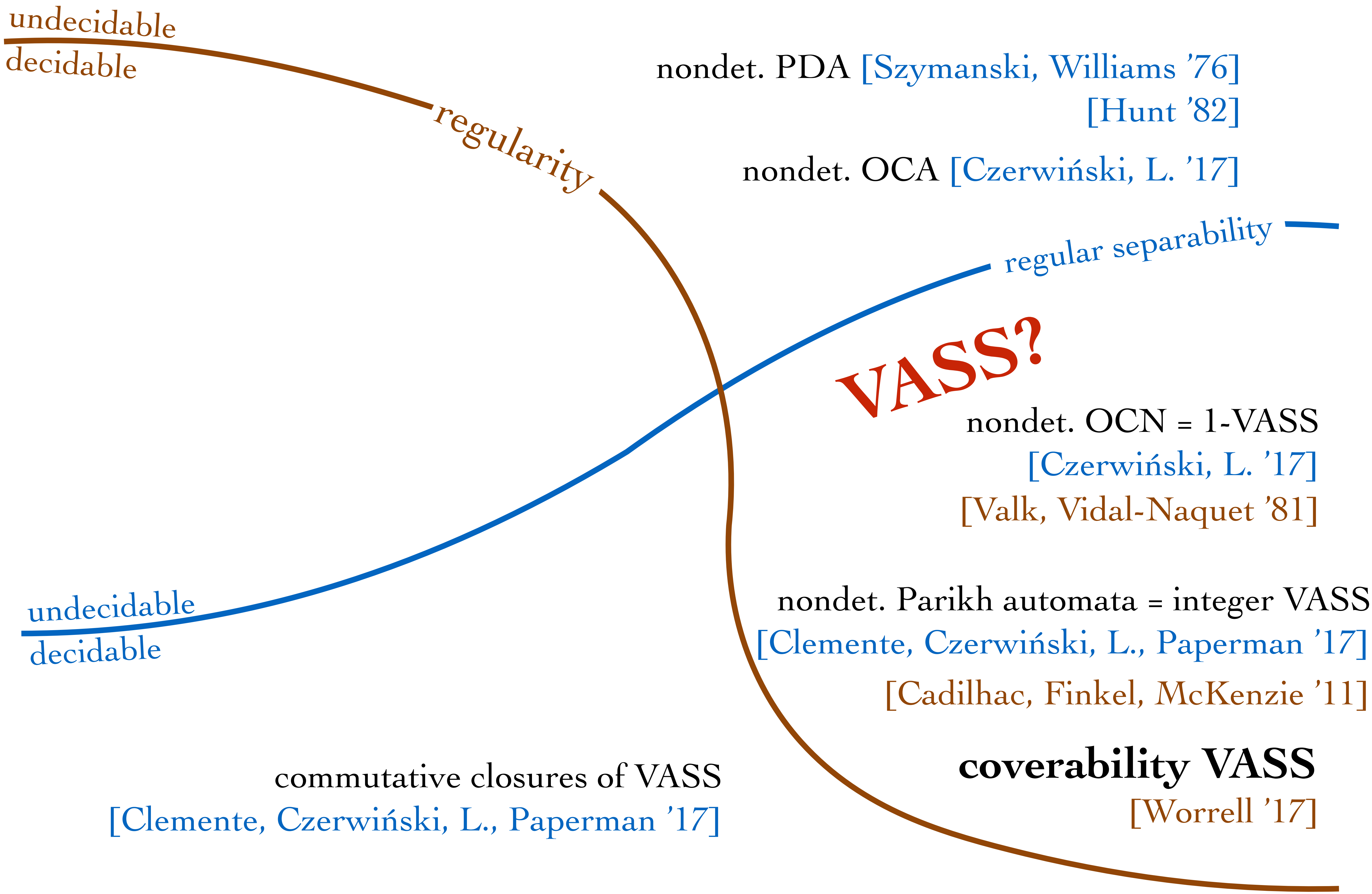
nondet. OCN = 1-VASS
[Czerwiński, L. ’17]

nondet. Parikh automata = integer VASS
[Clemente, Czerwiński, L., Paperman ’17]

commutative closures of VASS
[Clemente, Czerwiński, L., Paperman ’17]

coverability VASS

Regular separability as a decision problem



undecidable

decidable

regularity

nondet. PDA [Szymanski, Williams '76]
[Hunt '82]

nondet. OCA [Czerwiński, L. '17]

regular separability

VASS?

nondet. OCN = 1-VASS
[Czerwiński, L. '17]
[Valk, Vidal-Naquet '81]

undecidable

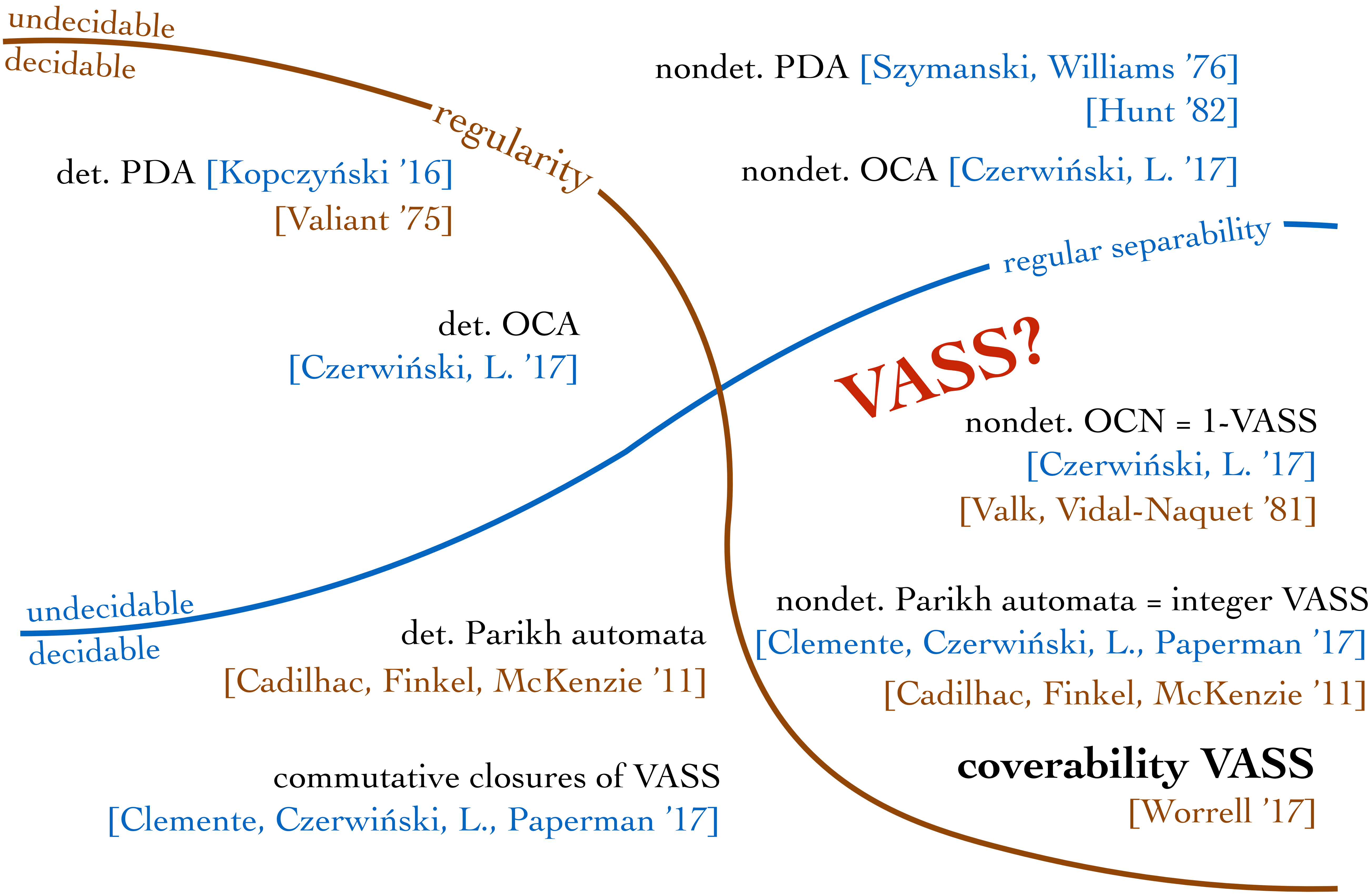
decidable

nondet. Parikh automata = integer VASS
[Clemente, Czerwiński, L., Paperman '17]
[Cadilhac, Finkel, McKenzie '11]

coverability VASS
[Worrell '17]

commutative closures of VASS
[Clemente, Czerwiński, L., Paperman '17]

Regular separability as a decision problem



undecidable

decidable

regularity

det. PDA [Kopczyński '16]
[Valiant '75]

det. OCA
[Czerwiński, L. '17]

undecidable

decidable

det. Parikh automata
[Cadilhac, Finkel, McKenzie '11]

commutative closures of VASS
[Clemente, Czerwiński, L., Paperman '17]

nondet. PDA [Szymanski, Williams '76]
[Hunt '82]

nondet. OCA [Czerwiński, L. '17]

regular separability

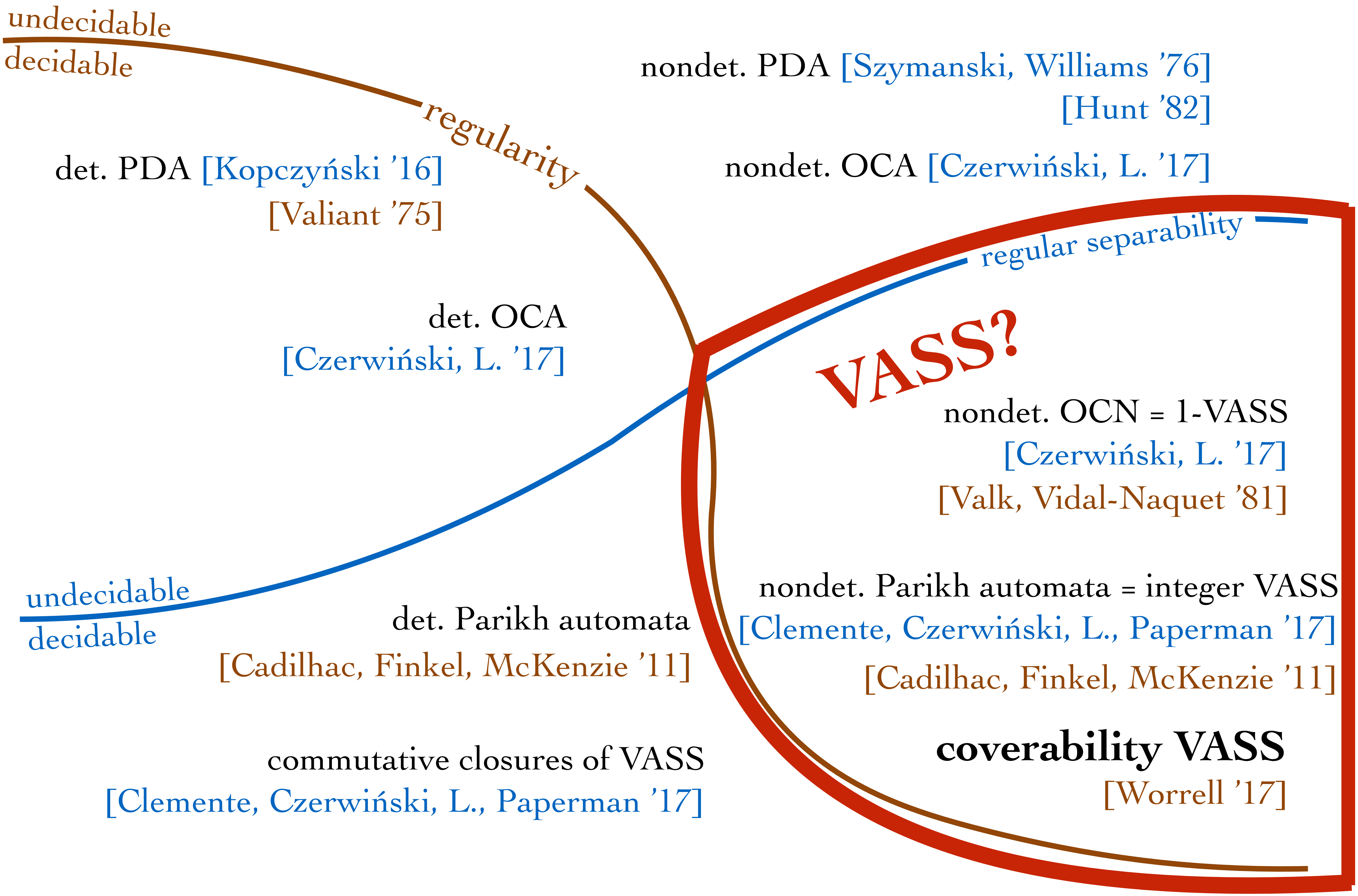
VASS?

nondet. OCN = 1-VASS
[Czerwiński, L. '17]
[Valk, Vidal-Naquet '81]

nondet. Parikh automata = integer VASS
[Clemente, Czerwiński, L., Paperman '17]
[Cadilhac, Finkel, McKenzie '11]

coverability VASS
[Worrell '17]

Regular separability as a decision problem



det. PDA [Kopczyński '16]
[Valiant '75]

det. OCA
[Czerwiński, L. '17]

det. Parikh automata
[Cadilhac, Finkel, McKenzie '11]

commutative closures of VASS
[Clemente, Czerwiński, L., Paperman '17]

nondet. PDA [Szymanski, Williams '76]
[Hunt '82]

nondet. OCA [Czerwiński, L. '17]

VASS?

nondet. OCN = 1-VASS
[Czerwiński, L. '17]
[Valk, Vidal-Naquet '81]

nondet. Parikh automata = integer VASS
[Clemente, Czerwiński, L., Paperman '17]
[Cadilhac, Finkel, McKenzie '11]

coverability VASS
[Worrell '17]