Parsing: Episode II

Matthew Might
University of Utah
matt.might.net
ucombinator.org

Administrivia

- Office hours: 10 am - 2pm (/ Lunch) Friday
- Project 1: The exponential late policy
- First friday frag hour: 2pm - 3pm Friday

Administrivia

- Office hours: 10 am - 2pm (/ Lunch) Friday
- Project 1: The exponential late policy
- First friday frag hour: 2pm - 3pm Friday

Administrivia

- Office hours: 10 am - 2pm (/ Lunch) Friday
- Project 1: The exponential late policy
- First friday frag hour: 2pm - 3pm Friday

Answers to last time

- Useful languages beyond LL(1)?
- Almost all of them: C, Java, ...
- Issue: `if x if y else z`
- Standard fix is to hack the parser

Today

- Push-down automata
- Push-down parsers
- CFG derivatives
- Parse strings
- Derivative parsers

Motivation

- Need to go beyond LL(k) or LR(*k*)
- No parser-generator available
- Need to parse ambiguous grammars

Learning goal

Be able to write YACC in less than 500 lines.

And, handle *any* CFG.

Push-down automaton

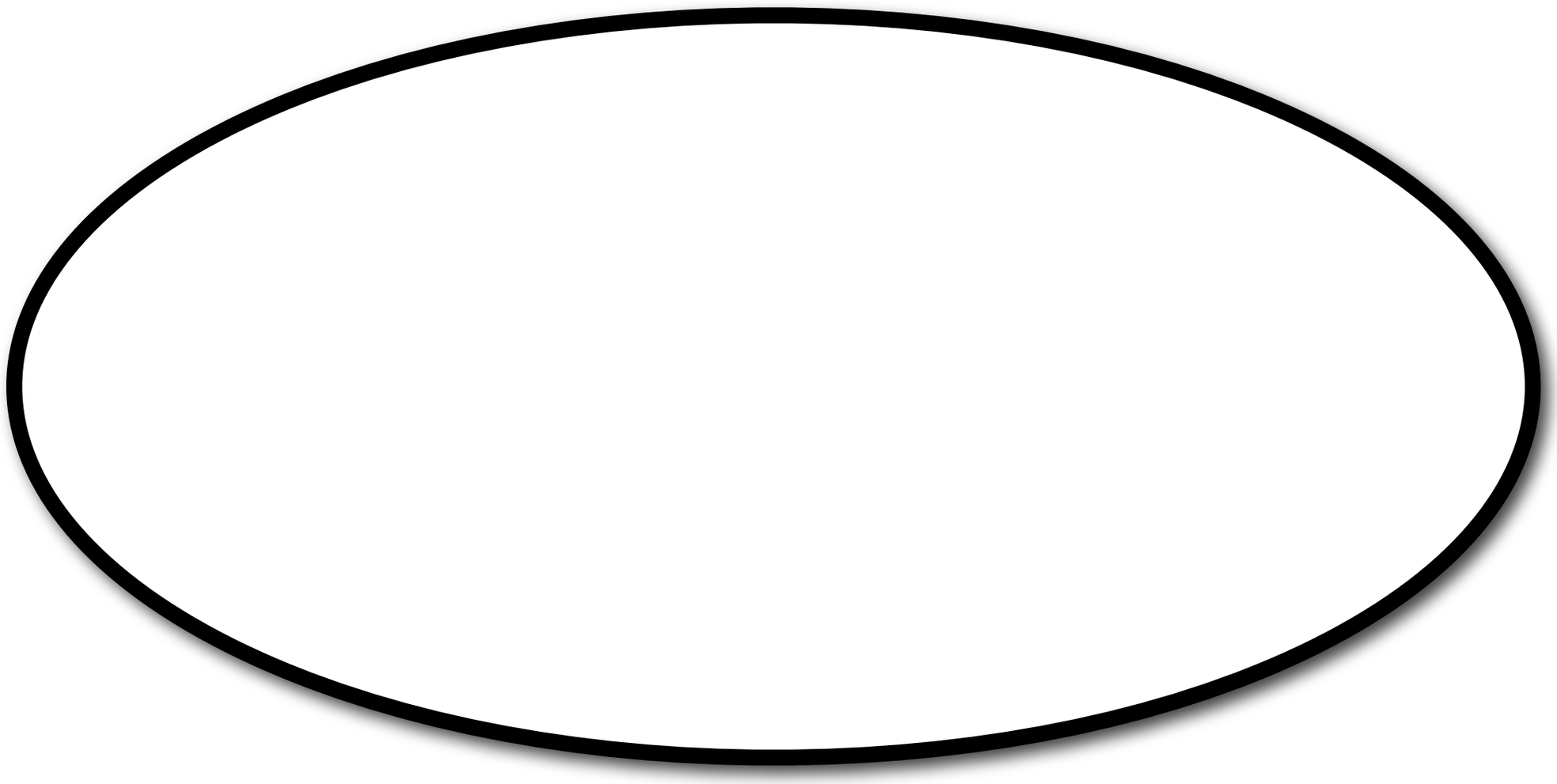
- Mechanical form of context-free language
- Given a string, it accepts or it does not*

PDA components

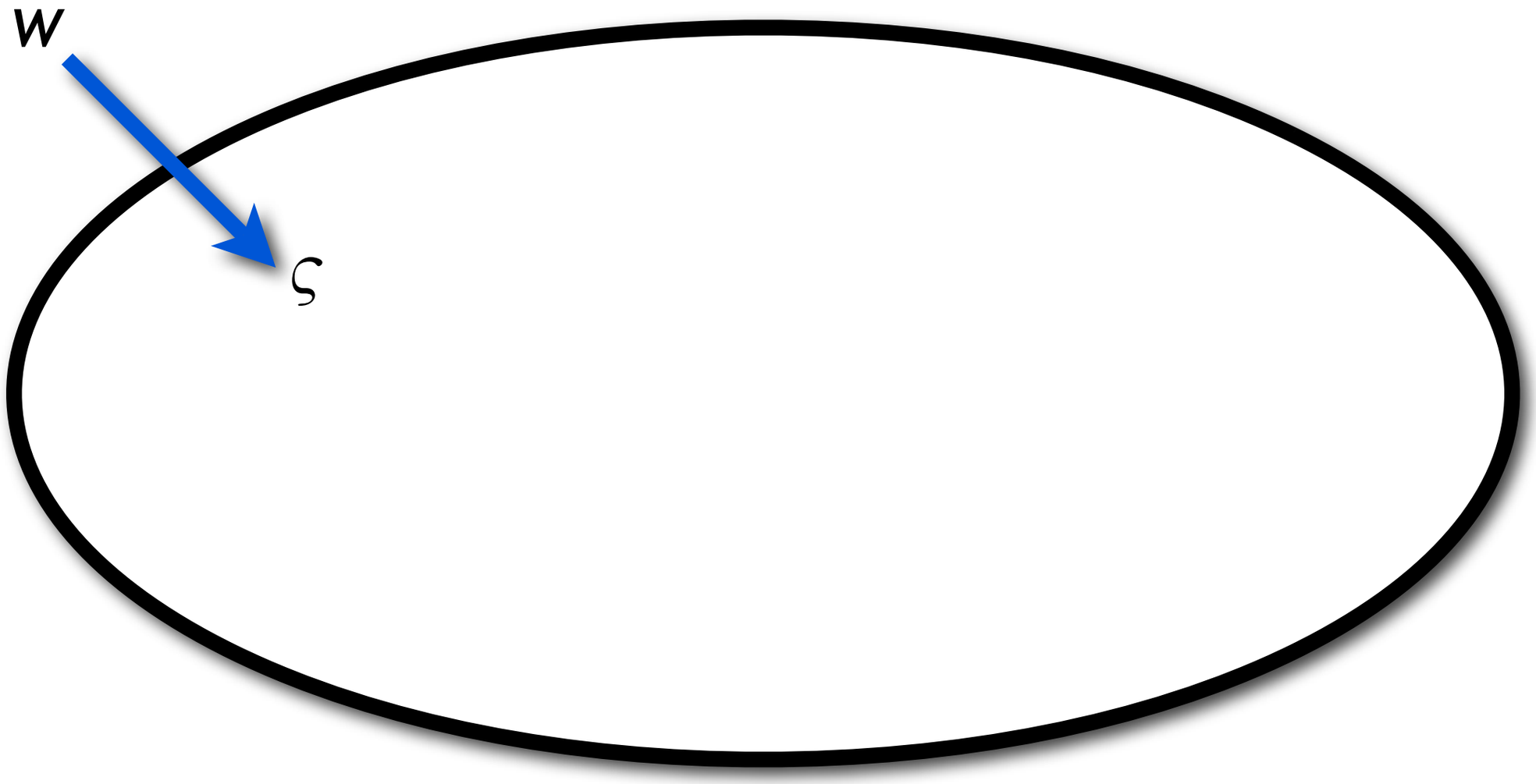
Formally, a **push-down autamaton** is a 7-tuple $(Q, A, \Gamma, \delta, q_0, \gamma_0, F)$, where

1. the set Q contains control states; and
2. the set A is the input alphabet; and
3. the set Γ is the stack alphabet; and
4. the relation $\delta \subseteq (Q \times (A \cup \{\epsilon\}) \times \Gamma) \times (Q \times \Gamma^*)$ is the transition relation; and
5. the control state $q_0 \in Q$ is the start state; and
6. the tape character $\gamma_0 \in \Gamma$ is the initial stack character; and
7. the states $F \subseteq Q$ are final/accepting states.

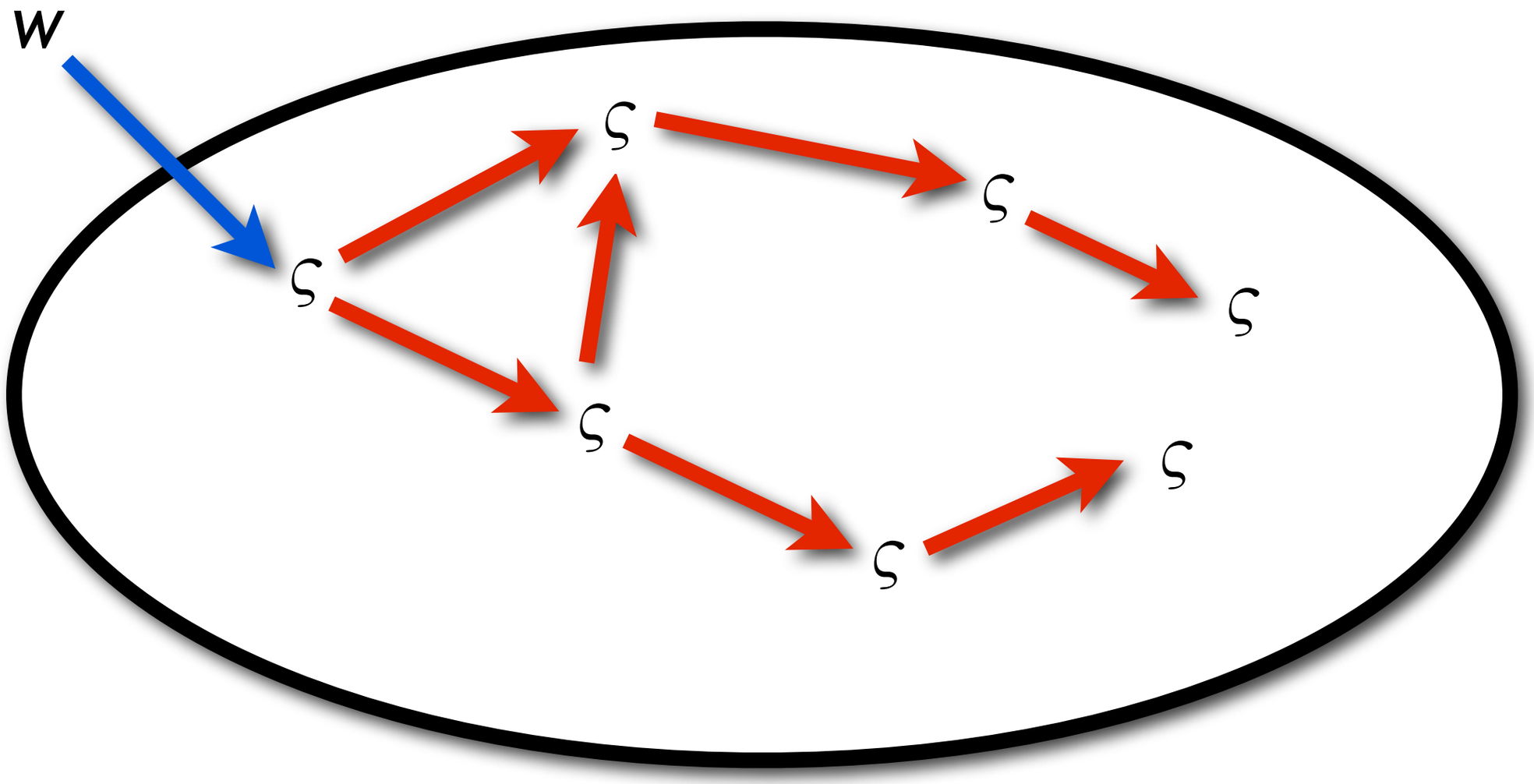
Recognition strategy



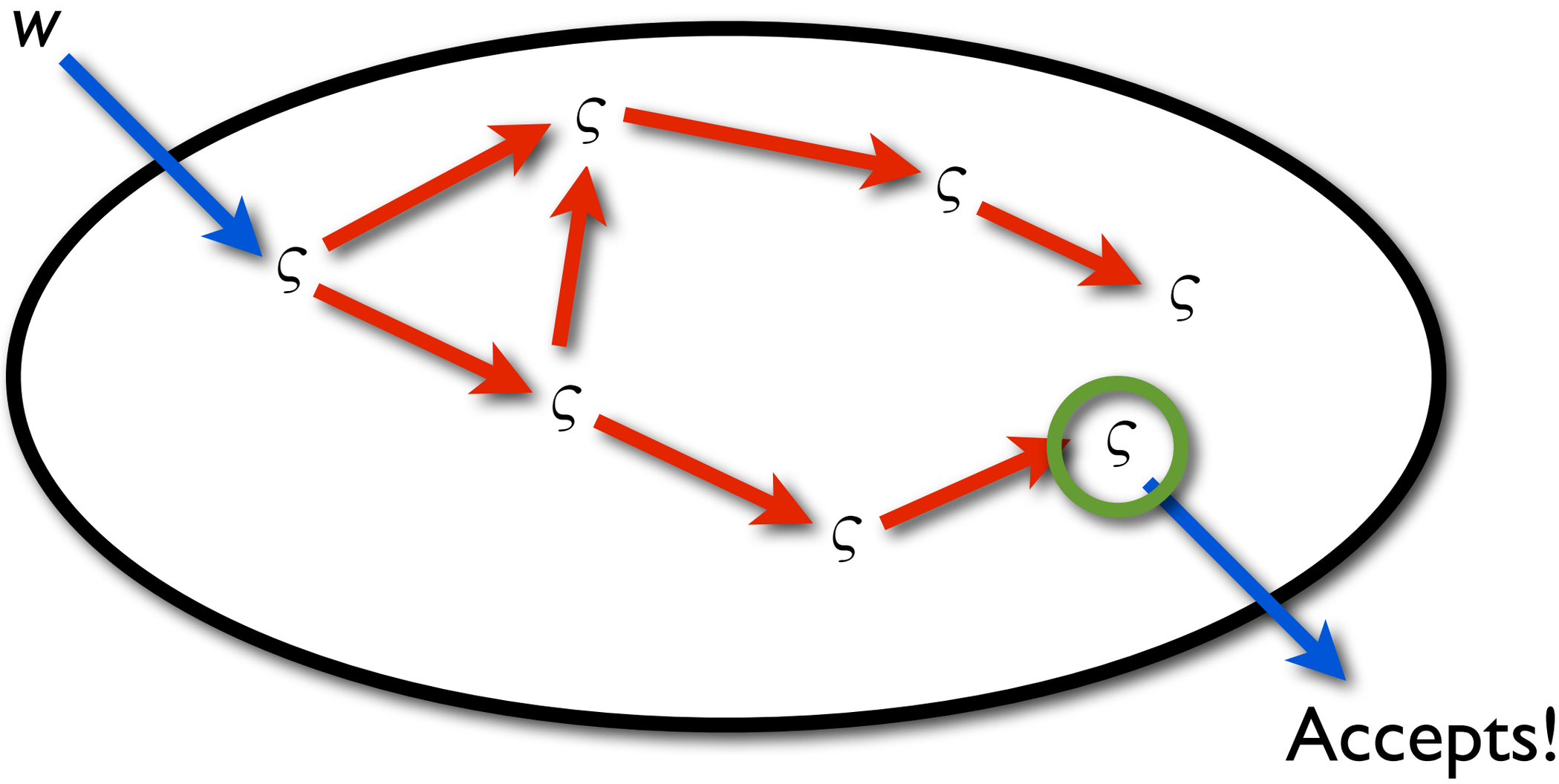
Recognition strategy



Recognition strategy



Recognition strategy



Configurations

A **machine configuration** is the complete state of a machine for some point in its execution.

Example: x86

- Value of every byte in RAM
- Values of all registers
- Values of CPU flags
- Value of program counter

Example: DFA

- Remaining input: *w*
- Current control-state: *q*

PDA configuration

$$\varsigma \in \Sigma = A^* \times \Gamma^* \times Q$$

PDA configuration

Configuration

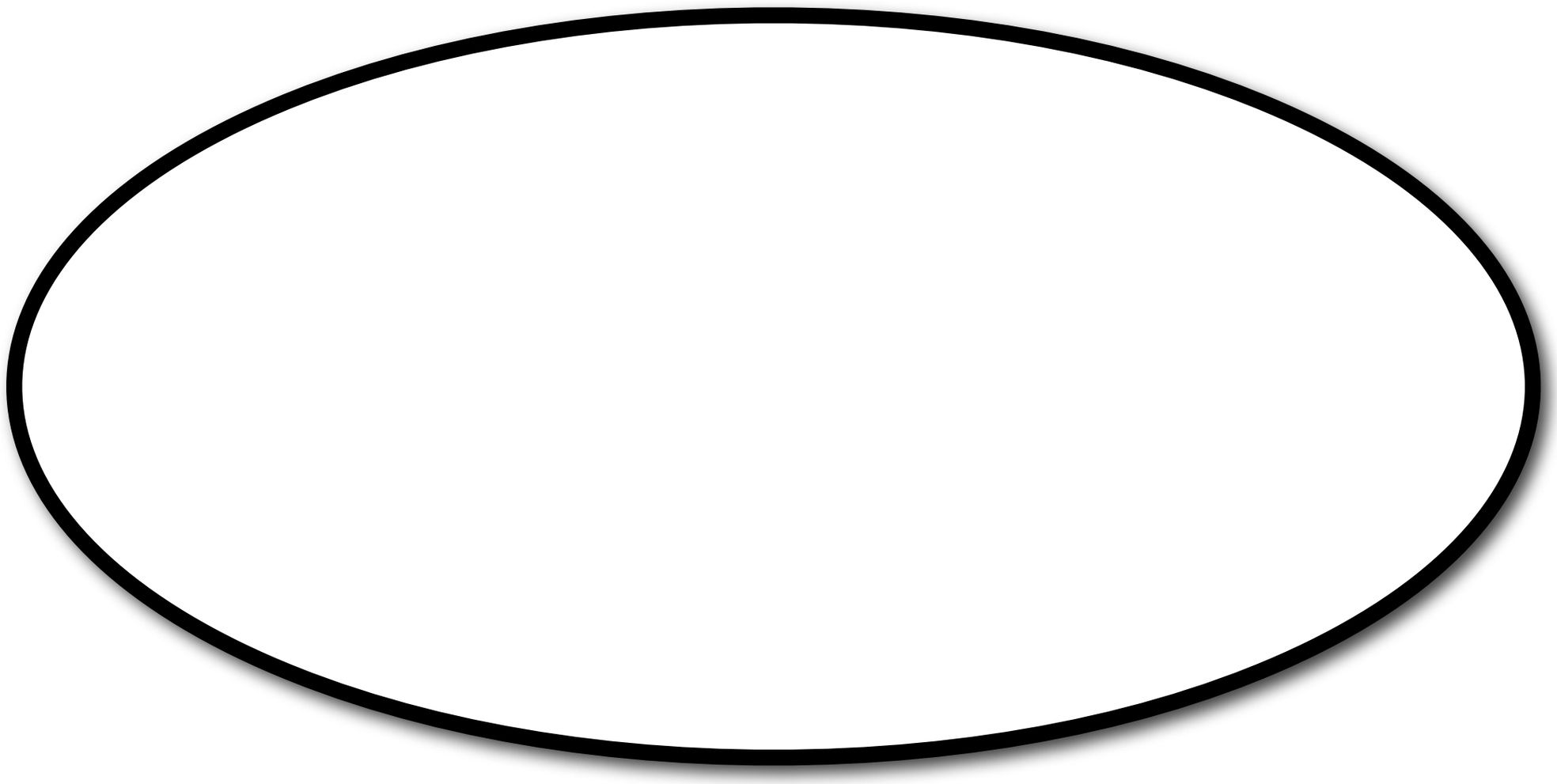
$$\varsigma \in \Sigma = A^* \times \Gamma^* \times Q$$

String Stack State

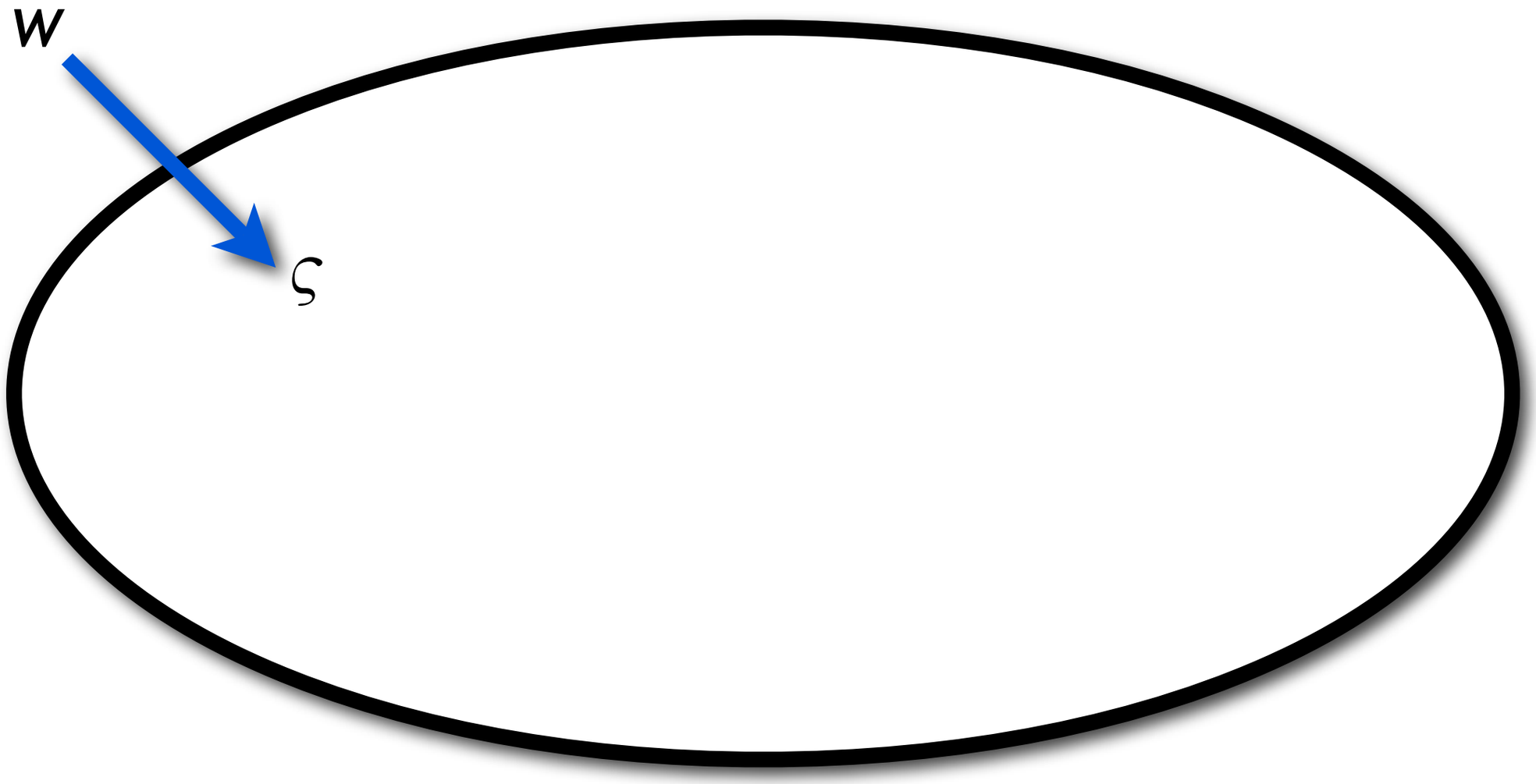
Initial configuration

$$\varsigma_0 = (w, \langle \gamma_0 \rangle, q_0)$$

Recognition strategy



Recognition strategy



Configuration transition

$$(\Rightarrow) \subseteq \Sigma \times \Sigma$$

$$(\Rightarrow) : \Sigma \longrightarrow \mathcal{P}(\Sigma)$$

Transition: Big idea

- Maybe consume an input character
- Maybe pop off the top stack character
- Maybe push several stack characters
- Maybe transition to a new control char
- Mix and match; PDA is nondeterministic

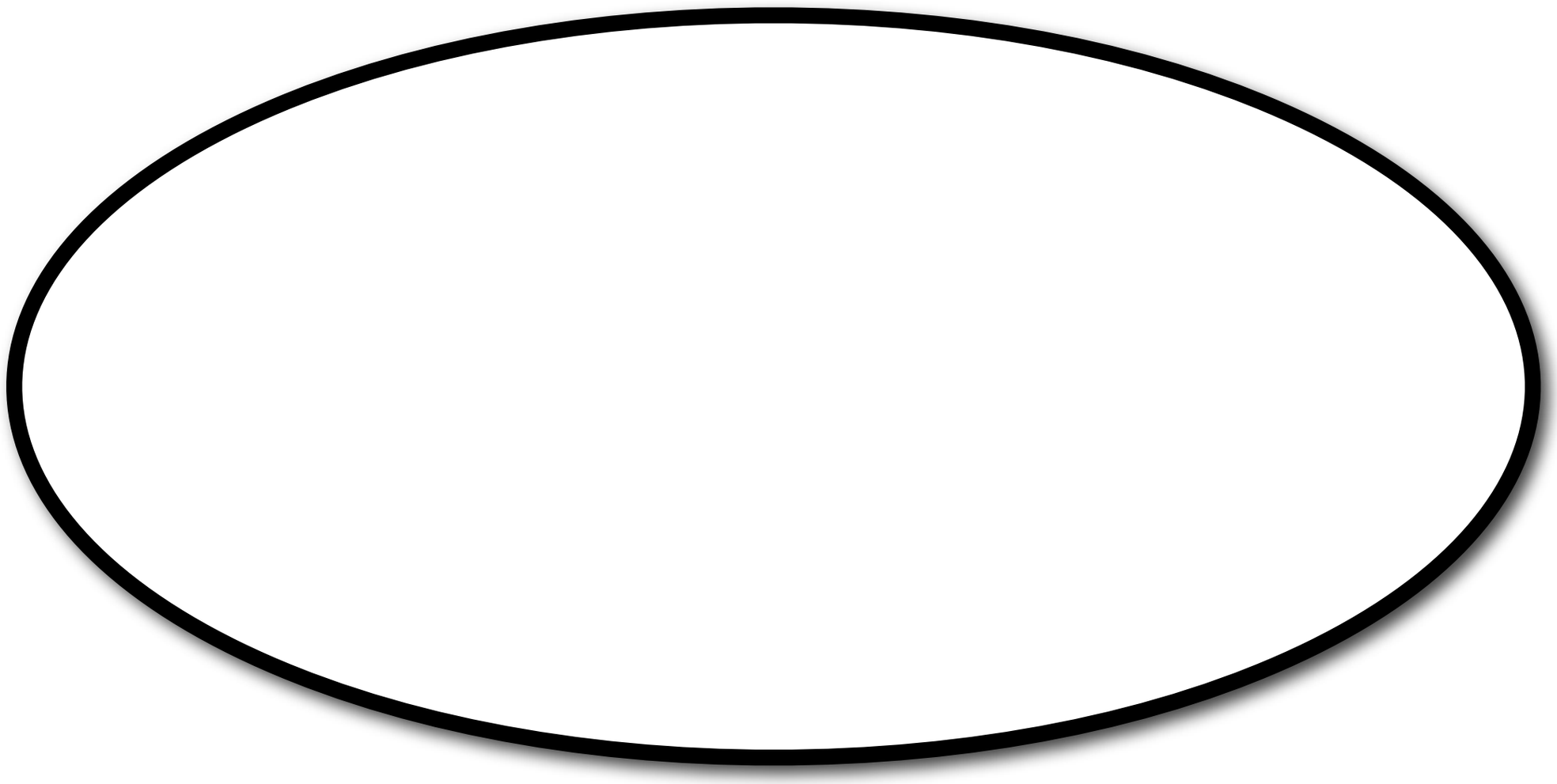
Configuration transition

$$\frac{(q, a, \gamma)\ \delta\ (q', \vec{\gamma}')}{(a : \vec{a}, \gamma : \vec{\gamma}, q) \Rightarrow^{M} (\vec{a}, \vec{\gamma}' +\!\!+ \vec{\gamma}, q'),}$$

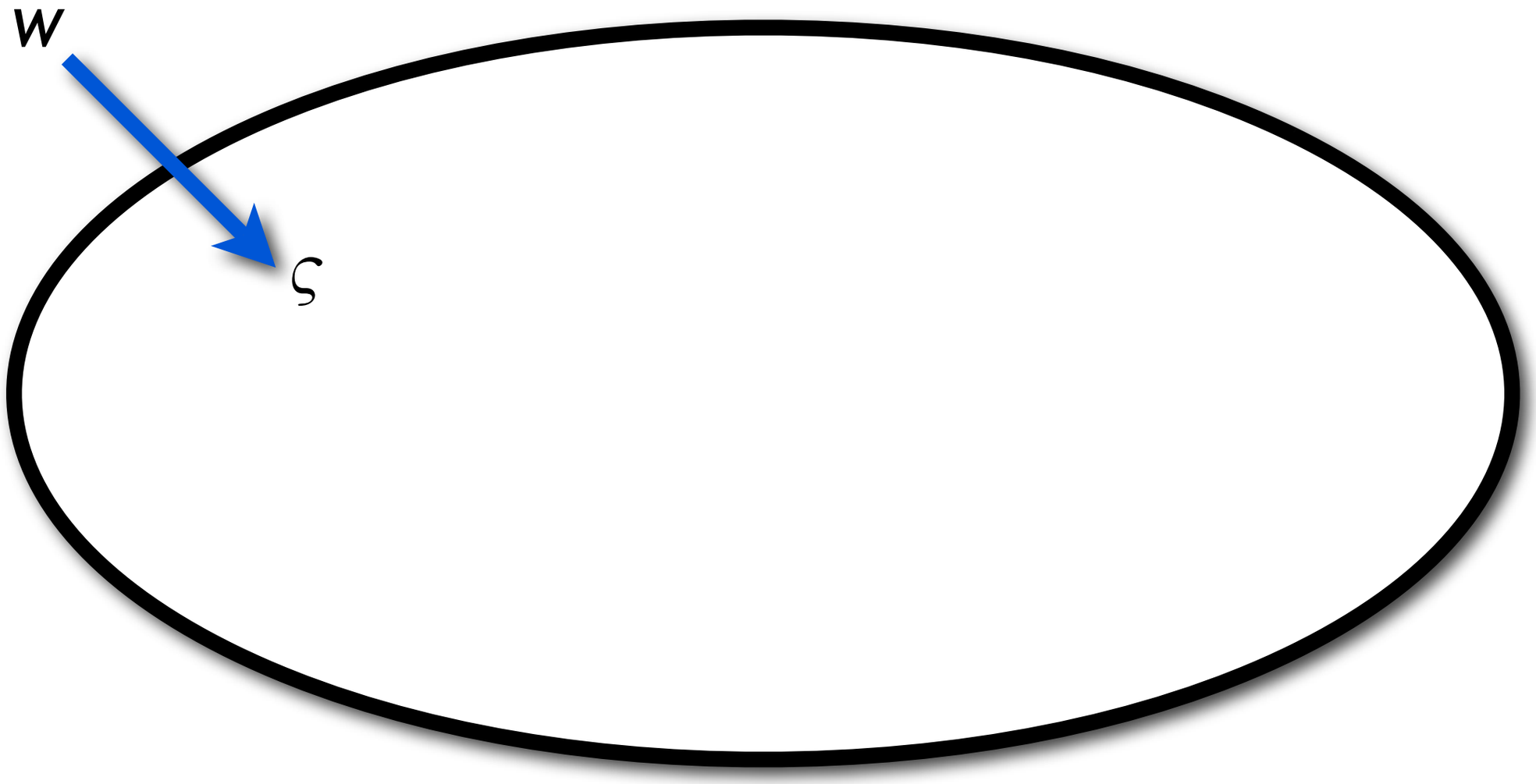
Configuration transition

$$\frac{(q, \epsilon, \gamma)\ \delta\ (q', \vec{\gamma}')}{(\vec{a}, \gamma : \vec{\gamma}, q) \Rightarrow^{M} (\vec{a}, \vec{\gamma}' +\!\!+ \vec{\gamma}, q').}$$

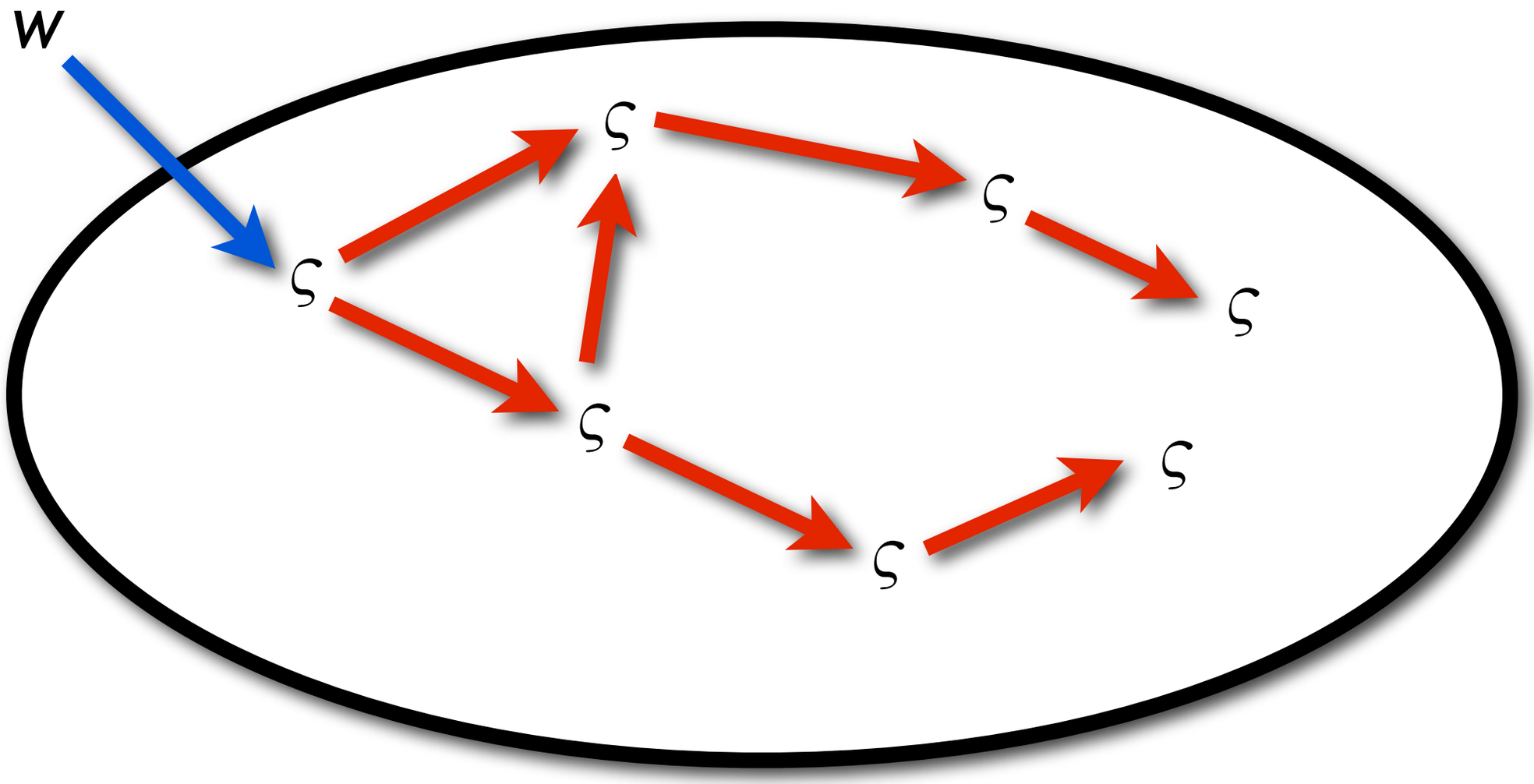
Recognition strategy



Recognition strategy



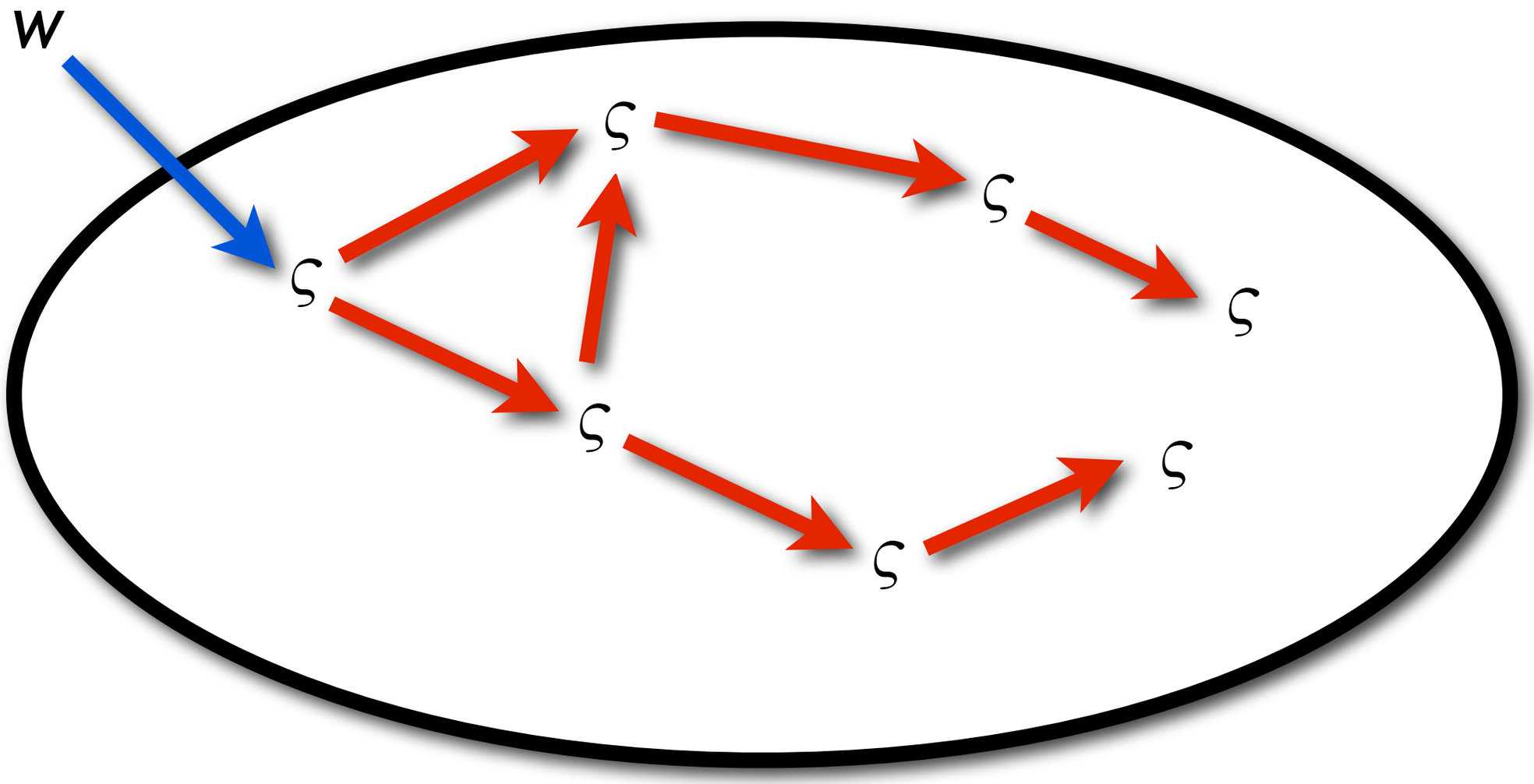
Recognition strategy



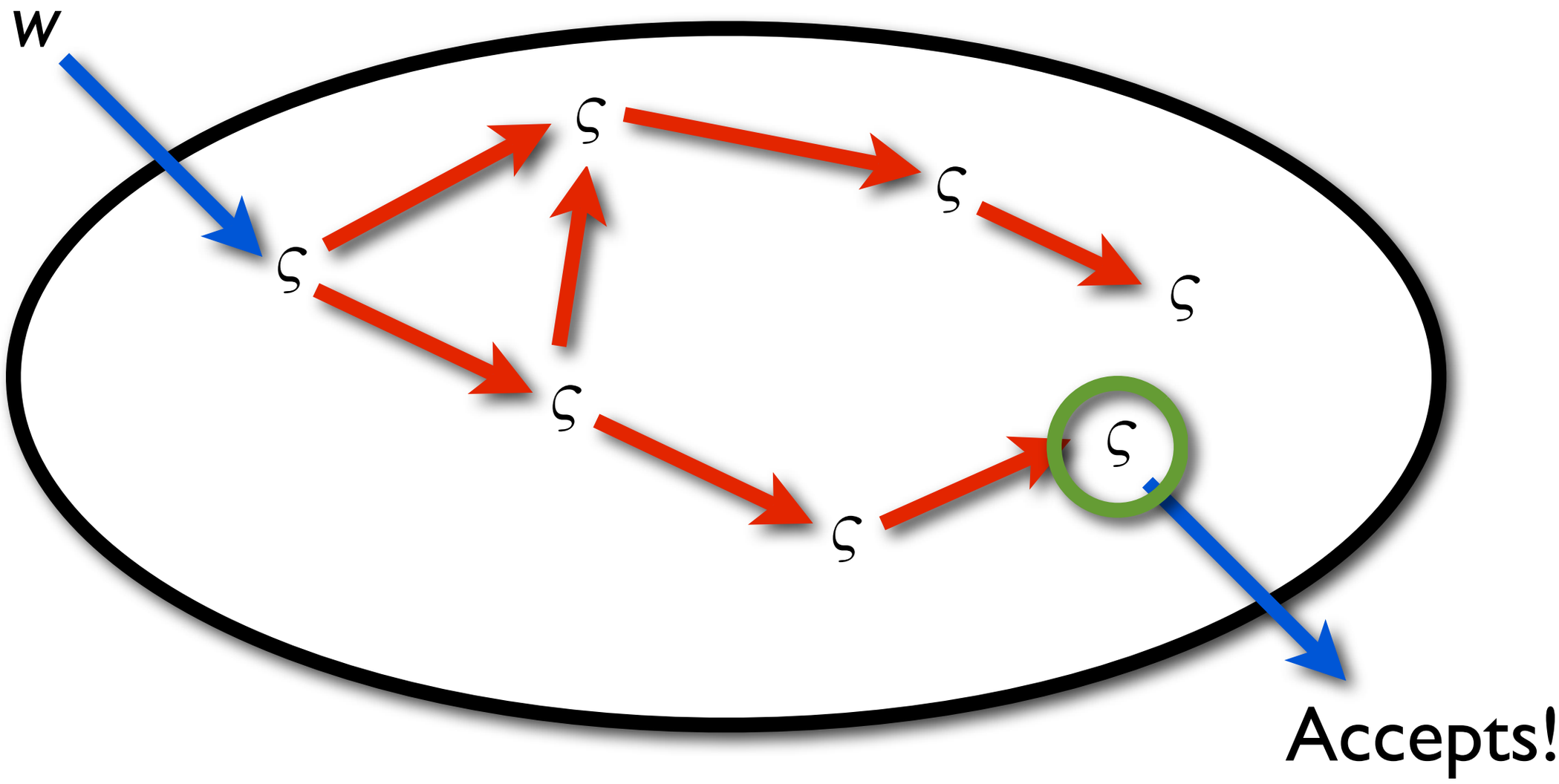
Checking acceptance

$$\frac{(w, \langle \gamma_0 \rangle, q_0) \Rightarrow^* (\epsilon, \vec{\gamma}', q') \qquad q' \in F}{w \in \mathcal{L}(Q, A, \Gamma, \delta, q_0, \gamma_0, F).}$$

Recognition strategy

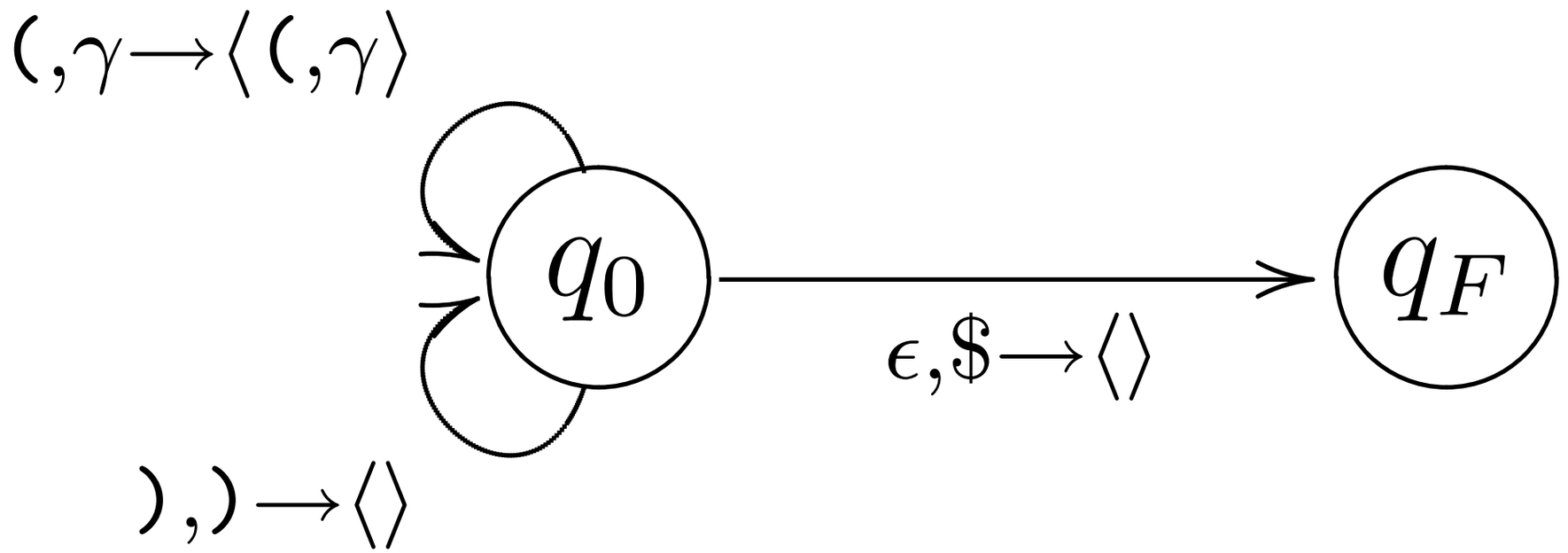


Recognition strategy



Example: Balanced ()

Balanced



Deterministic PDA

$$\delta_{\text{deterministic}} \in (Q \times A \times \Gamma) \rightarrow (Q \times \Gamma^*)$$

Intuition trivia

Intuition trivia

- Are NPDA and DPDA equivalent?

Intuition trivia

- Are NPDA and DPDA equivalent?
- Halting problem for deterministic PDA?

Intuition trivia

- Are NPDA and DPDA equivalent?
- Halting problem for deterministic PDA?
- Halting problem for nondeterministic PDA?

Intuition trivia

- Are NPDA and DPDA equivalent?
- Halting problem for deterministic PDA?
- Halting problem for nondeterministic PDA?
- Equality of two deterministic PDAs?

Intuition trivia

- Are NPDA and DPDA equivalent?
- Halting problem for deterministic PDA?
- Halting problem for nondeterministic PDA?
- Equality of two deterministic PDAs?
- Equality of two nondeterministic PDAs?

Recognizing CFGs

Big idea

- Use terminals + nonterminals for stack
- Start with the CFG's starting symbol
- If nonterminal on top, replace with a rule
- If terminal on top, pop it and consume it

CFG => PDA

$$(A, N, R, n_0)$$

$$(Q, A, \Gamma, \delta, q_0, \gamma_0, F)$$

CFG => PDA

$$Q = \{q_0, q_{\text{match}}, q_F\}$$

$$\Gamma = A \cup N \cup \{\$\}$$

$$\gamma_0 = \$.$$

$$F = \{q_F\}.$$

CFG => PDA

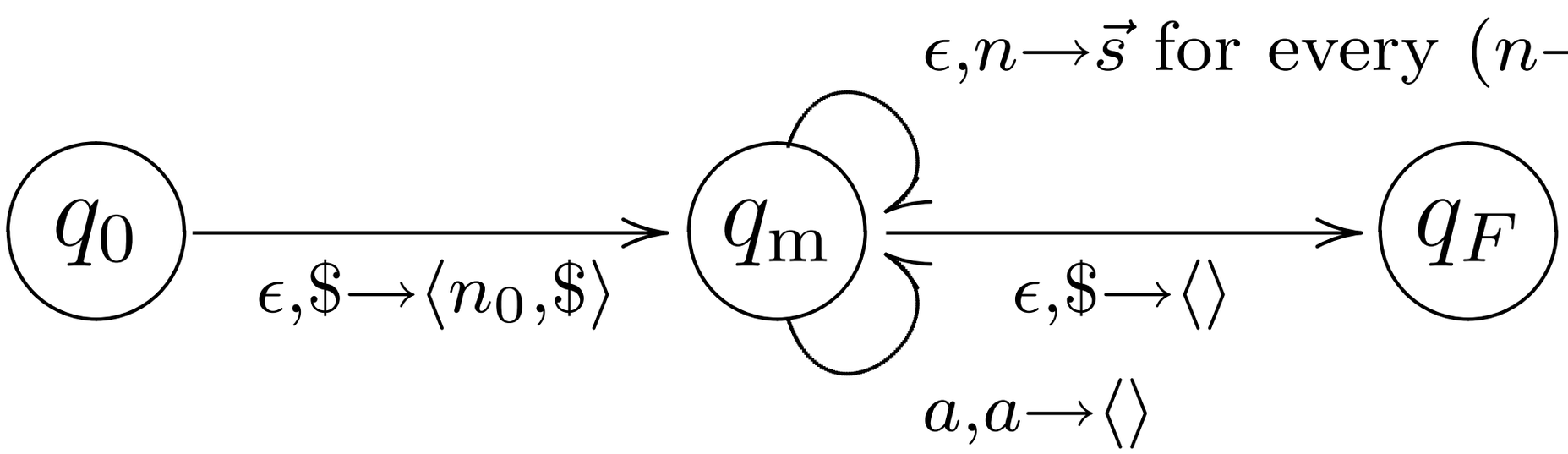
$$(q_0, \epsilon, \$)\ \delta\ (q_{\text{match}}, \langle n_0, \$ \rangle)$$

$$(q_{\text{match}}, \epsilon, n)\ \delta\ (q_{\text{match}}, \langle s_1 \dots s_m \rangle) \text{ if } (n \to s_1 \dots s_m) \in R$$

$$(q_{\text{match}}, a, a)\ \delta\ (q_{\text{match}}, \langle\rangle)$$

$$(q_{\text{match}}, \epsilon, \$)\ \delta\ (q_F, \langle\rangle).$$

CFG => PDA

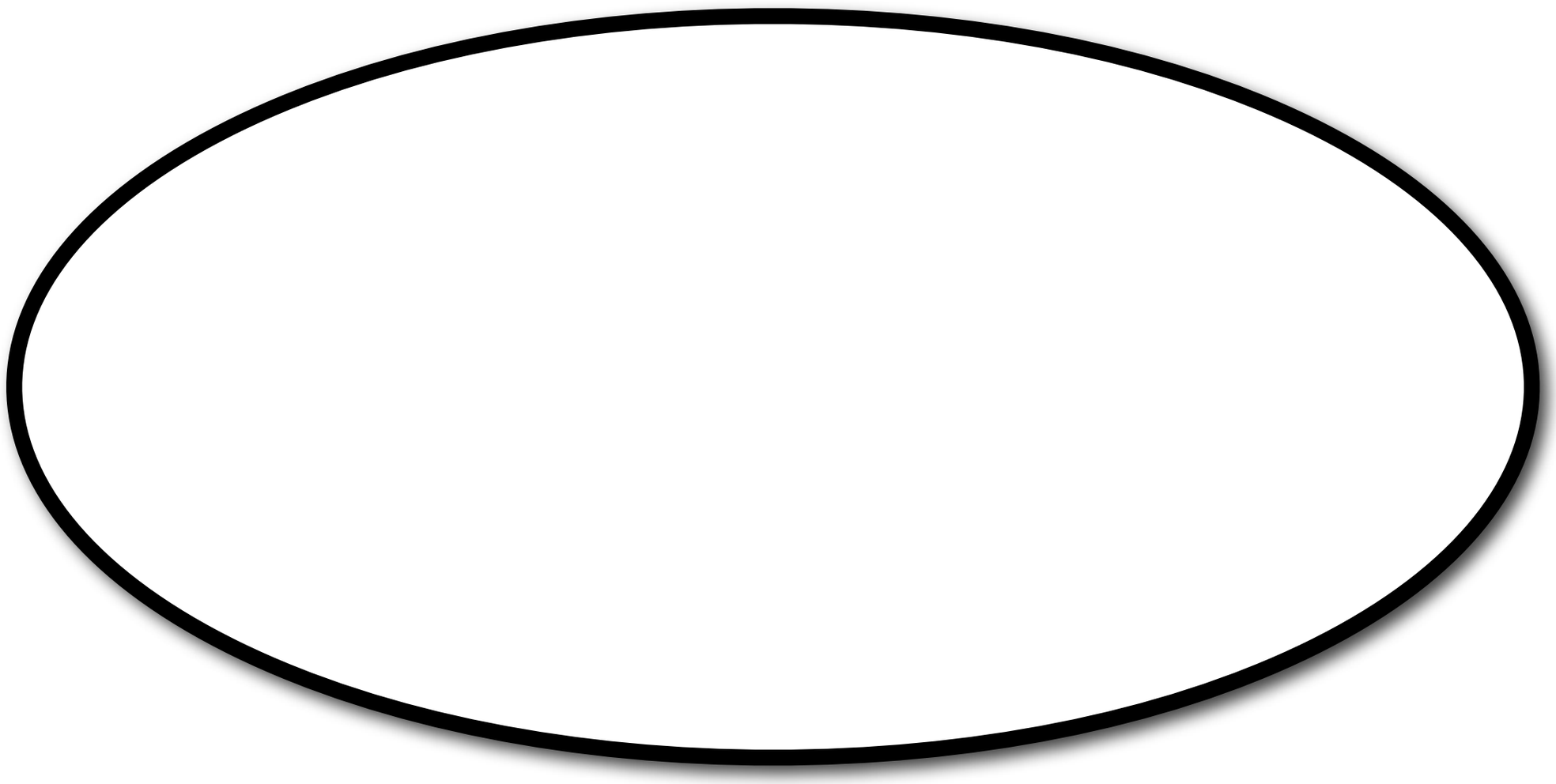

$\epsilon,n\rightarrow\vec{s}$ for every $(n\rightarrow\vec{s})\in R$
q_0
$\epsilon,\$\rightarrow\langle n_0,\$\rangle$
q_m
$\epsilon,\$\rightarrow\langle\rangle$
q_F
$a,a\rightarrow\langle\rangle$

Optimizations

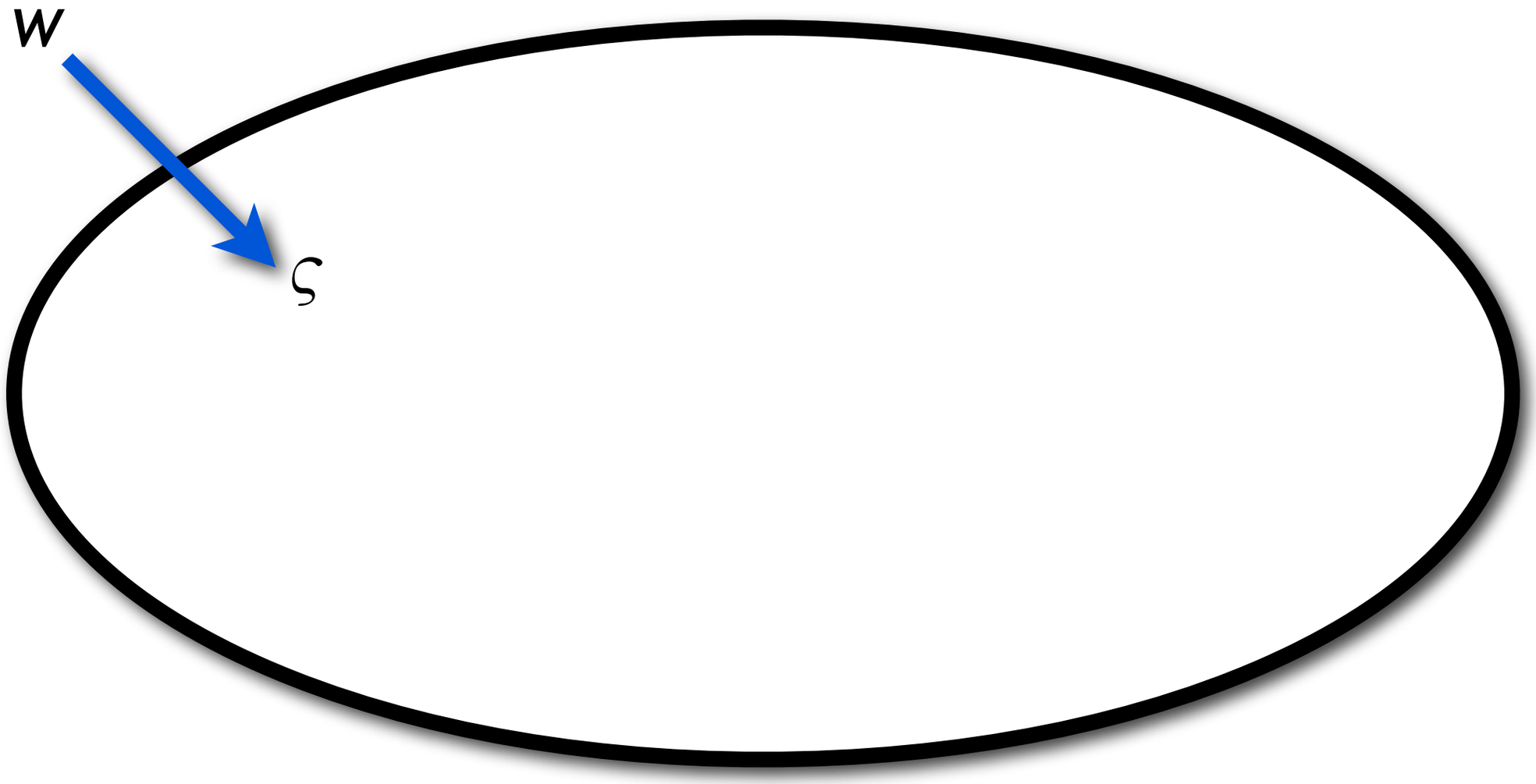
- Use first sets
- Breadth-first

Parsing

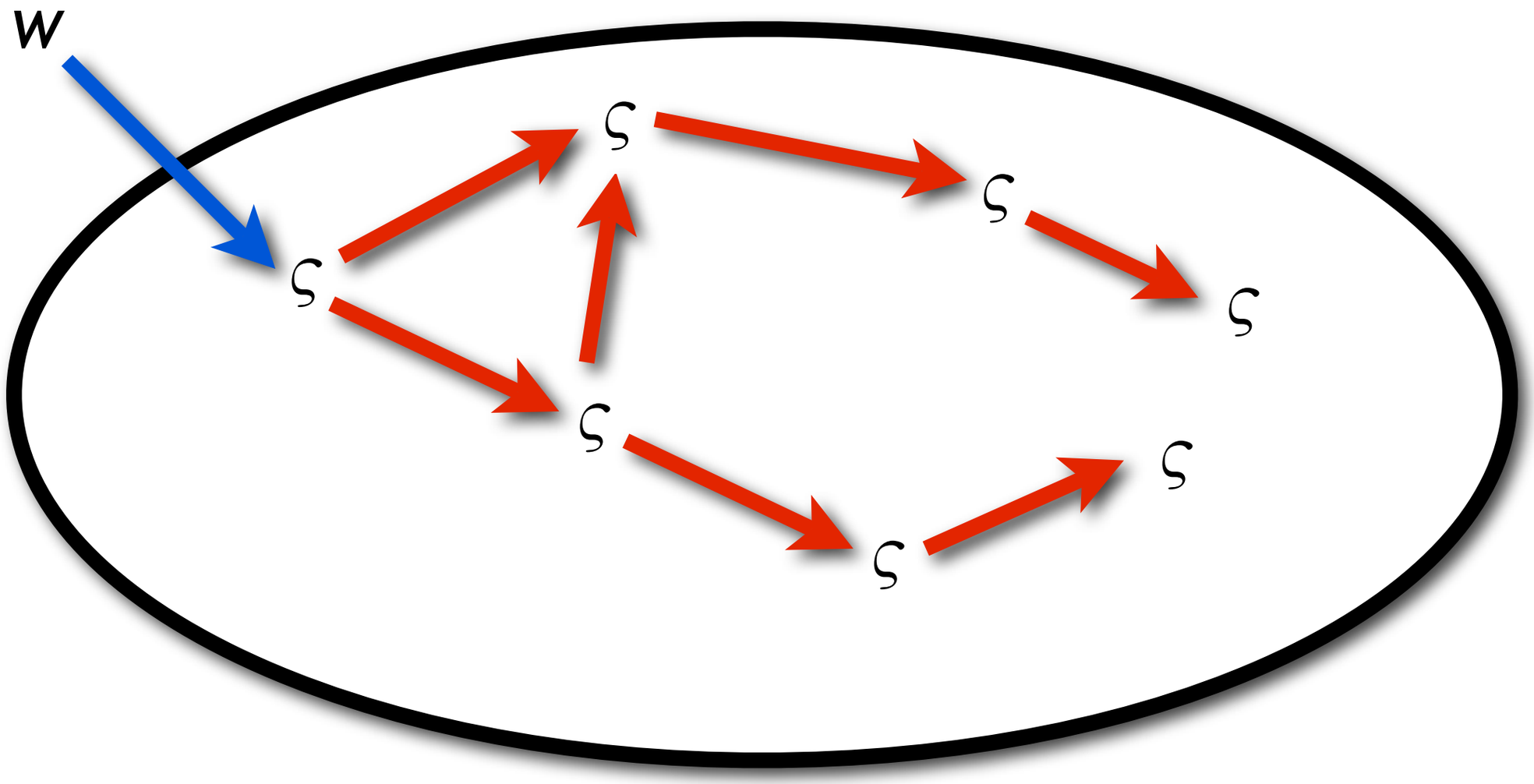
Parsing strategy



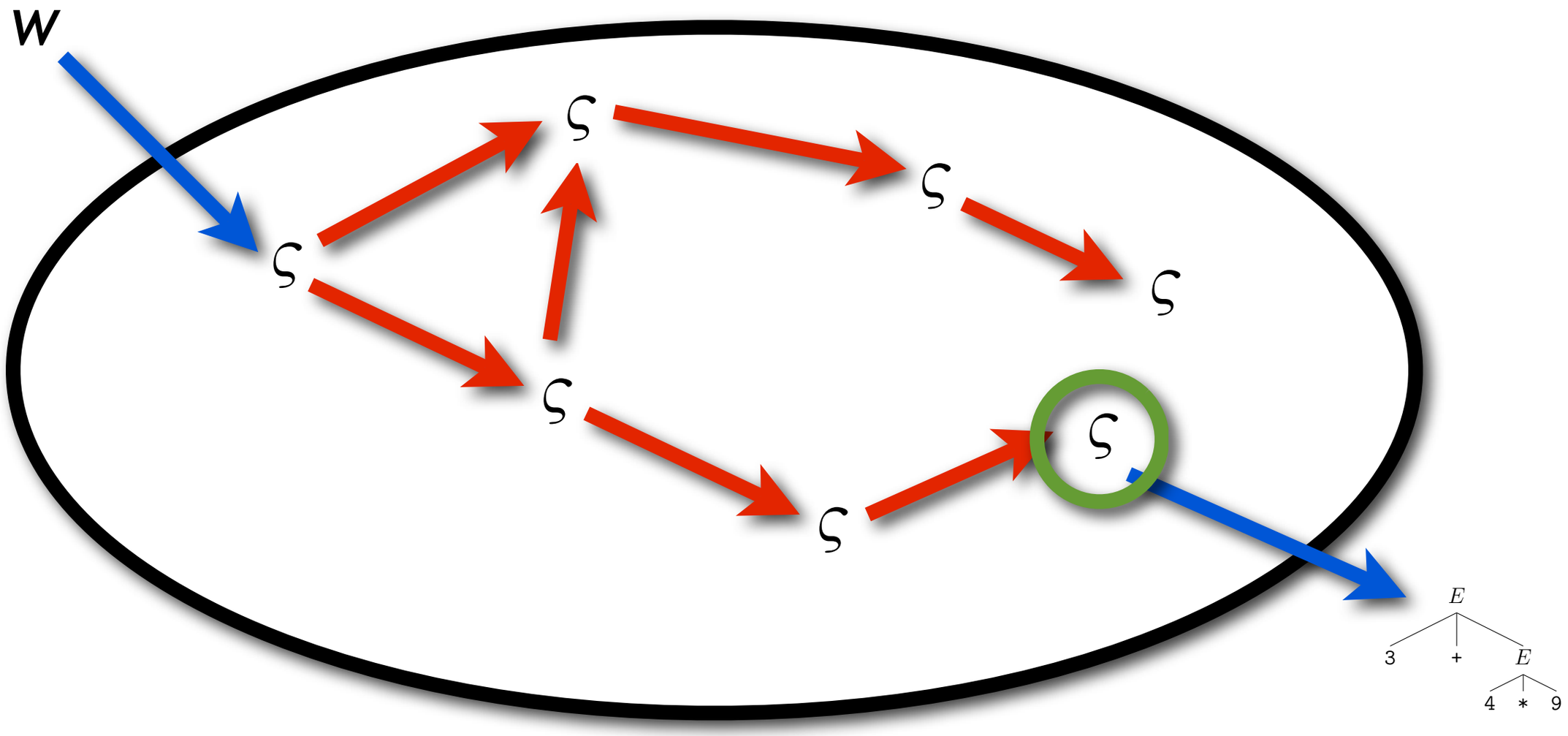
Parsing strategy



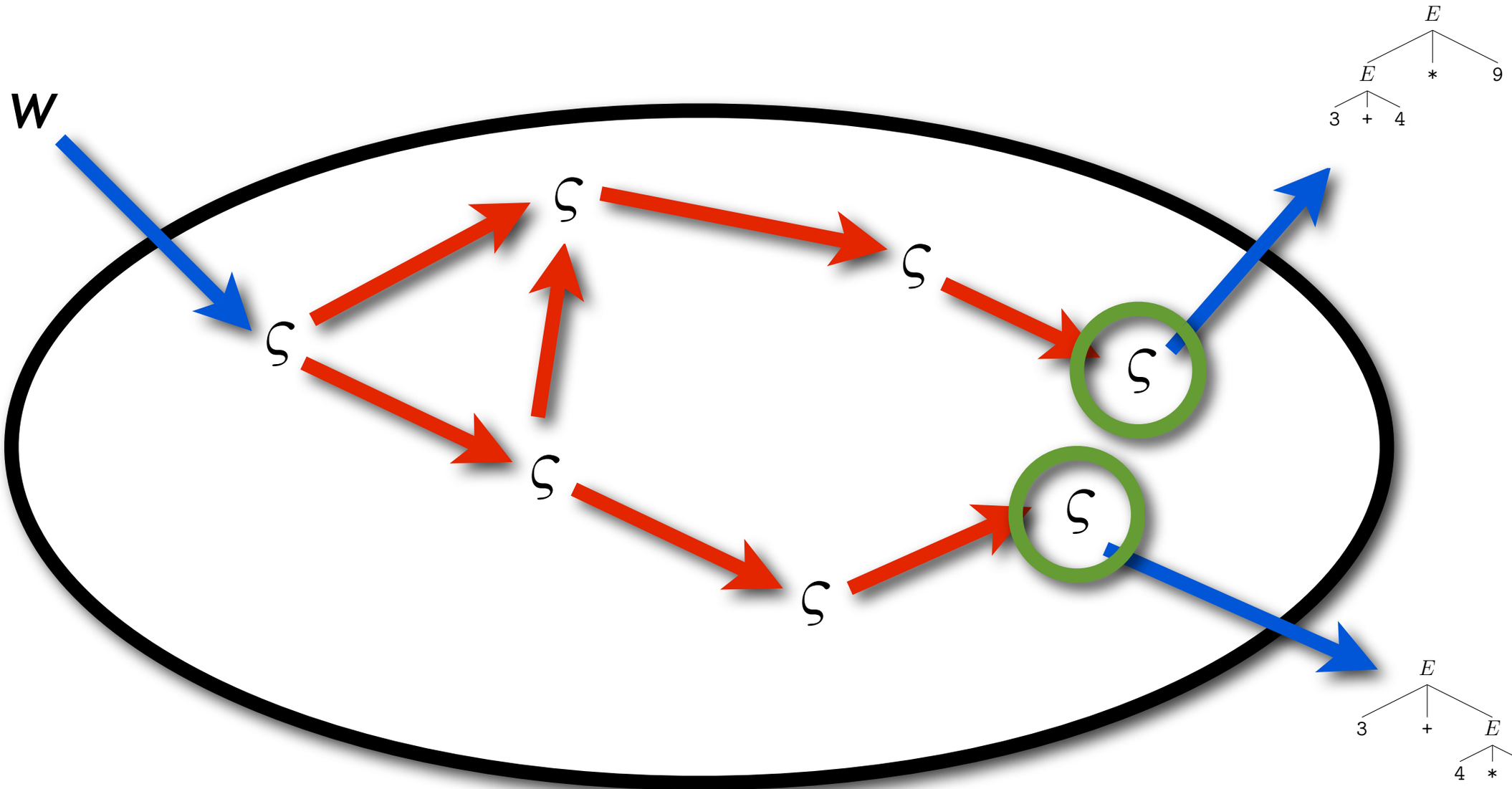
Parsing strategy



Parsing strategy



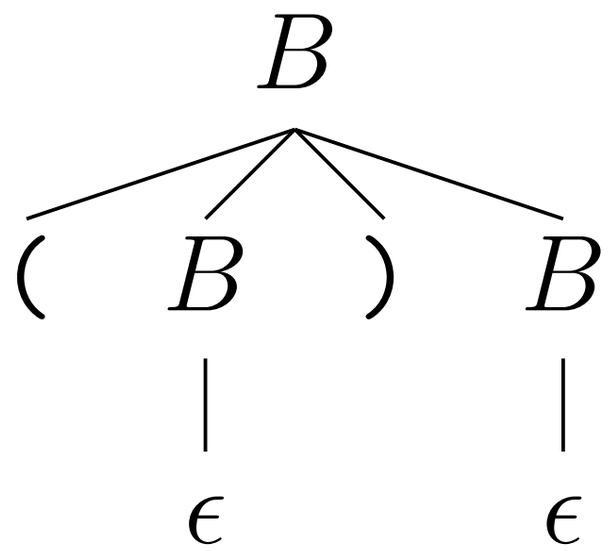
Parsing strategy



Parse trees

$$t \in T = A + (R \times T^*)$$

Example: ()



$$(B \to (B)\,B, \langle (, (B \to \epsilon, \langle\rangle),), (B \to \epsilon, \langle\rangle)\rangle)$$

Strategy

- Include reduction rule hints in PDA
- Construct the tree in bottom-up fashion

CFG => PDA => Tree

$$Q = \{q_0, q_\mathrm{m}, q_F\}$$

$$\Gamma = A \cup N \cup R \cup \{\$\}$$

$$\gamma_0 = \$.$$

$$F = \{q_F\}.$$

CFG => PDA => Tree

$$
\begin{aligned}
(q_0, \epsilon, \$)\ \delta\ (q_{\mathrm{m}}, \langle n_0, \$ \rangle) \\
(q_{\mathrm{m}}, \epsilon, n)\ \delta\ (q_{\mathrm{m}}, \langle s_1, \ldots, s_m, (n \longrightarrow s_1 \ldots s_m) \rangle)\ \text{if}\ (n \longrightarrow s_1 \ldots s_m) \in R \\
(q_{\mathrm{m}}, a, a)\ \delta\ (q_{\mathrm{m}}, \langle \rangle) \\
(q_{\mathrm{m}}, \epsilon, \$)\ \delta\ (q_F, \langle \rangle) \\
(q_{\mathrm{m}}, \epsilon, n \longrightarrow s_1 \ldots s_m)\ \delta\ (q_{\mathrm{m}}, \langle \rangle).
\end{aligned}
$$

CFG => PDA => Tree

$$(q_0, \epsilon, \$)\ \delta\ (q_\mathrm{m}, \langle n_0, \$ \rangle)$$

$$(q_\mathrm{m}, \epsilon, n)\ \delta\ (q_\mathrm{m}, \langle s_1, \ldots, s_m, (n \longrightarrow s_1 \ldots s_m) \rangle)\ \text{if}\ (n \longrightarrow s_1 \ldots s_m) \in R$$

$$(q_\mathrm{m}, a, a)\ \delta\ (q_\mathrm{m}, \langle \rangle)$$

$$(q_\mathrm{m}, \epsilon, \$)\ \delta\ (q_F, \langle \rangle)$$

$$(q_\mathrm{m}, \epsilon, n \longrightarrow s_1 \ldots s_m)\ \delta\ (q_\mathrm{m}, \langle \rangle)$$

Pop-labeled transition

$$(\Rightarrow) \subseteq \Sigma \times \Gamma^? \times \Sigma$$

Example

$$\varsigma \Longrightarrow^{a} \varsigma'$$

Parsing machine

$$\psi \in \Psi = \Sigma \times T^*$$

$$(\rightarrowtriangle) \subseteq \Psi \times \Psi$$

Rule I

$$\frac{\varsigma \Rightarrow^{a} \varsigma'}{(\varsigma, \vec{t}) \rightarrowtriangle (\varsigma', a : \vec{t}).}$$

Rule 2

$$\frac{\varsigma \Rightarrow^{n \to s_1 \ldots s_m} \varsigma'}{(\varsigma, \langle t_m, \ldots, t_1 \rangle +\!\!+ \vec{t}) \to (\varsigma', (n \to s_1 \ldots s_m, \langle t_1, \ldots, t_m \rangle) : \vec{t}).}$$

Rule 3

$$\frac{\varsigma \Rightarrow^{n} \varsigma' \text{ or } \varsigma \Rightarrow^{\epsilon} \varsigma'}{(\varsigma, \vec{t}) \rightarrowtriangle (\varsigma', \vec{t}).}$$

Derivatives, reloaded

WARNING:

RESEARCH AHEAD

Derivatives

$$D_c L = \{w : cw \in L\}$$

Example

$$D_{\mathtt{f}}\{\mathtt{foo}, \mathtt{frak}, \mathtt{bar}\} = \{\mathtt{oo}, \mathtt{rak}\}$$

Derivative of CFL

$$D_c(\mathcal{L}(G)) = \mathcal{L}(A, N', R', n_0')$$

Nonterminals

$$N' = N \cup \{D_c(n) : n \in N\}$$

Rules

for each rule $(n \rightarrow s_1 \dots s_m)$, if the sequence $s_1 \dots s_i$ is nullable, then:

$$[D_c(n) \rightarrow D_c(s_{i+1}) s_{i+2} \dots s_m] \in R',$$

and if $m = 0$, then :

$$D_c(n \rightarrow \emptyset),$$

and

$$(n \rightarrow s_1 \dots s_m) \in R'.$$

Start symbol

$$n_0' = D_c(n_0)$$

Matching

$$\frac{w \in D_c(L)}{cw \in L.}$$

Optimizing derivative

- Compute D_c lazily -- when derived, firsted
- Short-circuit computation once $\varnothing$ appears

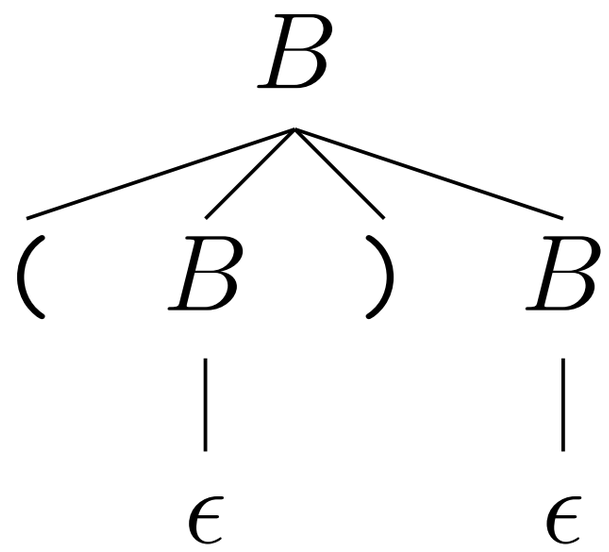
Parsing strategy

- Augment language to emit parse strings

Parse strings

- Post-order rendering of a parse tree
- Encodes information to rebuild parse tree

Example



$$([B \to \epsilon])[B \to \epsilon][B \to (B)\,B].$$

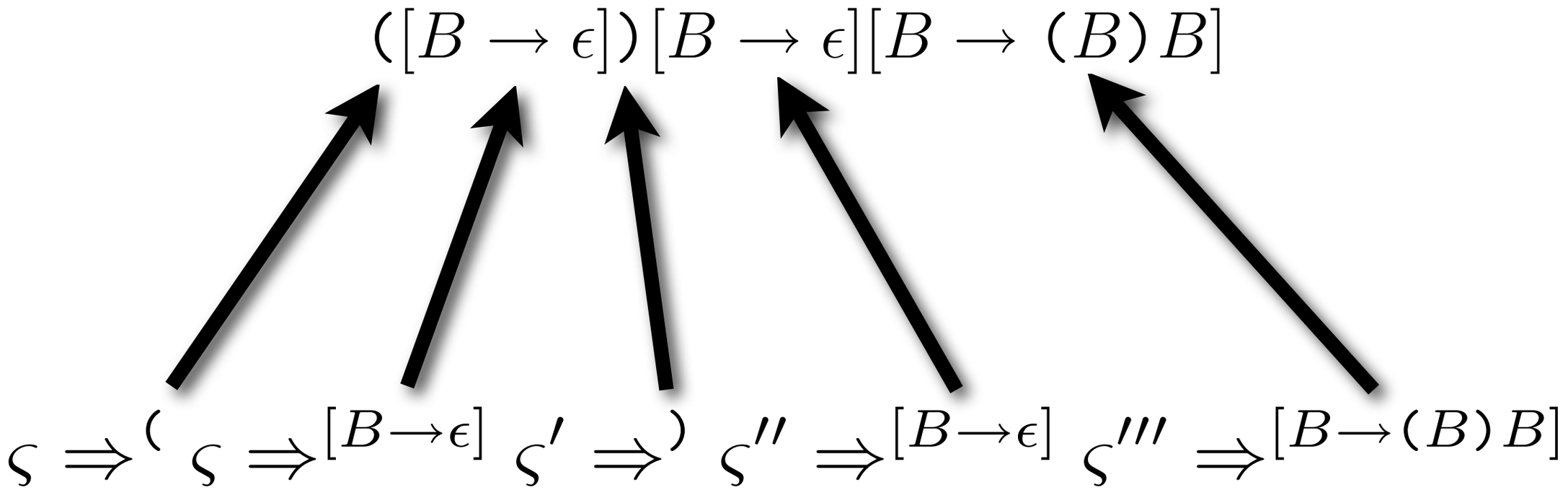
Parsing machine

$$\frac{\varsigma \Rightarrow^{a} \varsigma'}{(\varsigma, \overrightarrow{t}) \rightarrowtriangle (\varsigma', a : \overrightarrow{t}).}$$

$$\frac{\varsigma \Rightarrow^{n \rightarrow s_1 \ldots s_m} \varsigma'}{(\varsigma, \langle t_m, \ldots, t_1 \rangle +\!\!+ \overrightarrow{t}) \rightarrowtriangle (\varsigma', (n \rightarrow s_1 \ldots s_m, \langle t_1, \ldots, t_m \rangle) : \overrightarrow{t}).}$$

Example

Example


$([B \to \epsilon])[B \to \epsilon][B \to (B)\,B]$
$\varsigma \Rightarrow^{(} \varsigma \Rightarrow^{[B\to\epsilon]} \varsigma' \Rightarrow^{)} \varsigma'' \Rightarrow^{[B\to\epsilon]} \varsigma''' \Rightarrow^{[B\to(B)\,B]}$

Derivative machine

$$G = (A, N, R, n_0)$$

$$G' = (A \cup R, N, R', n_0)$$

Configurations

$$\varsigma \in \Sigma = A^* \times \mathbb{L}_{\mathsf{CF}}(A \cup R)$$

Shift

$$(aw, L) \Rightarrow^{a} (w, D_c(L))$$

Reduce

$$\frac{r \in \mathit{first}(L)}{(w, L) \Rightarrow^r (w, D_r(L)),}$$

Intuition trivia

- Derivative of context-sensitive closed?
- Derivative of Turing-recognizable possible?

Project questions?