JISET: JavaScript IR-based Semantics Extraction Toolchain

Jihyeok Park, Jihee Park, Seungmin An, Sukyoung Ryu

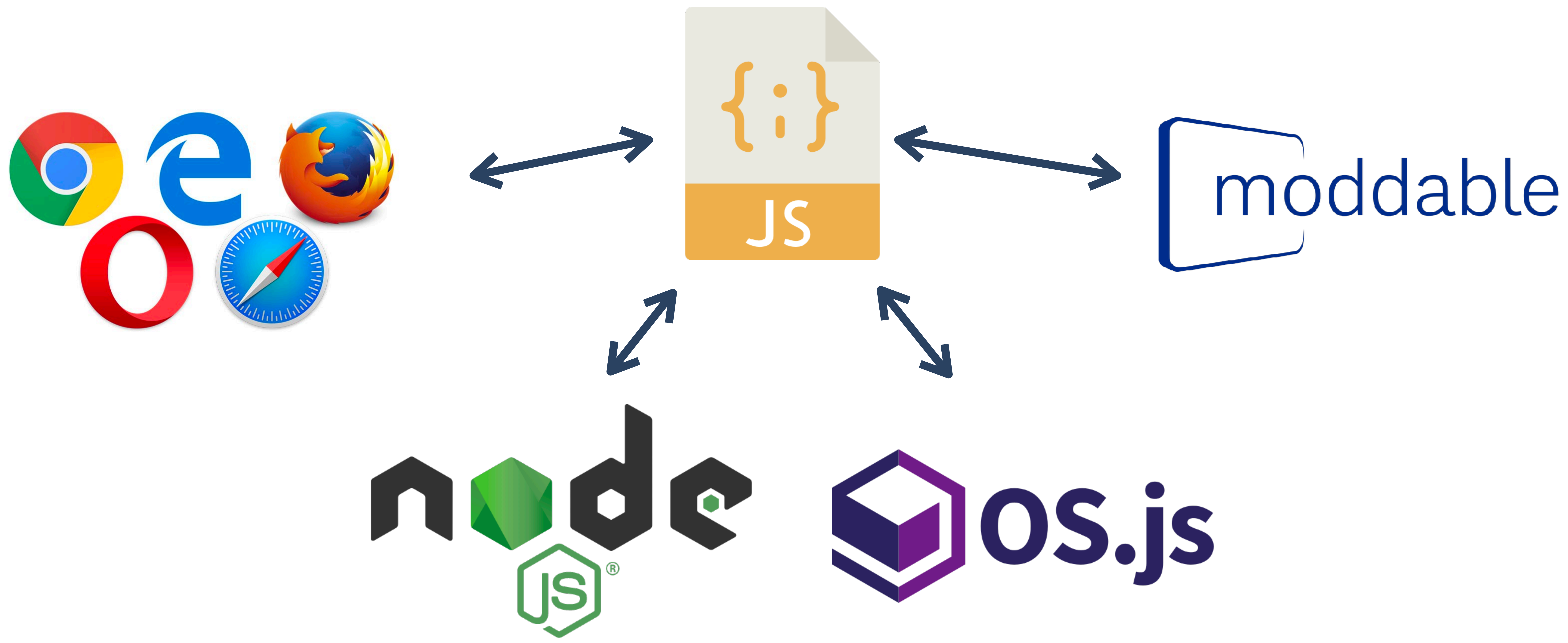
PLRG @ KAIST

In Proceedings of the 35th IEEE/ACM International Conference on Automated Software Engineering (ASE'20)

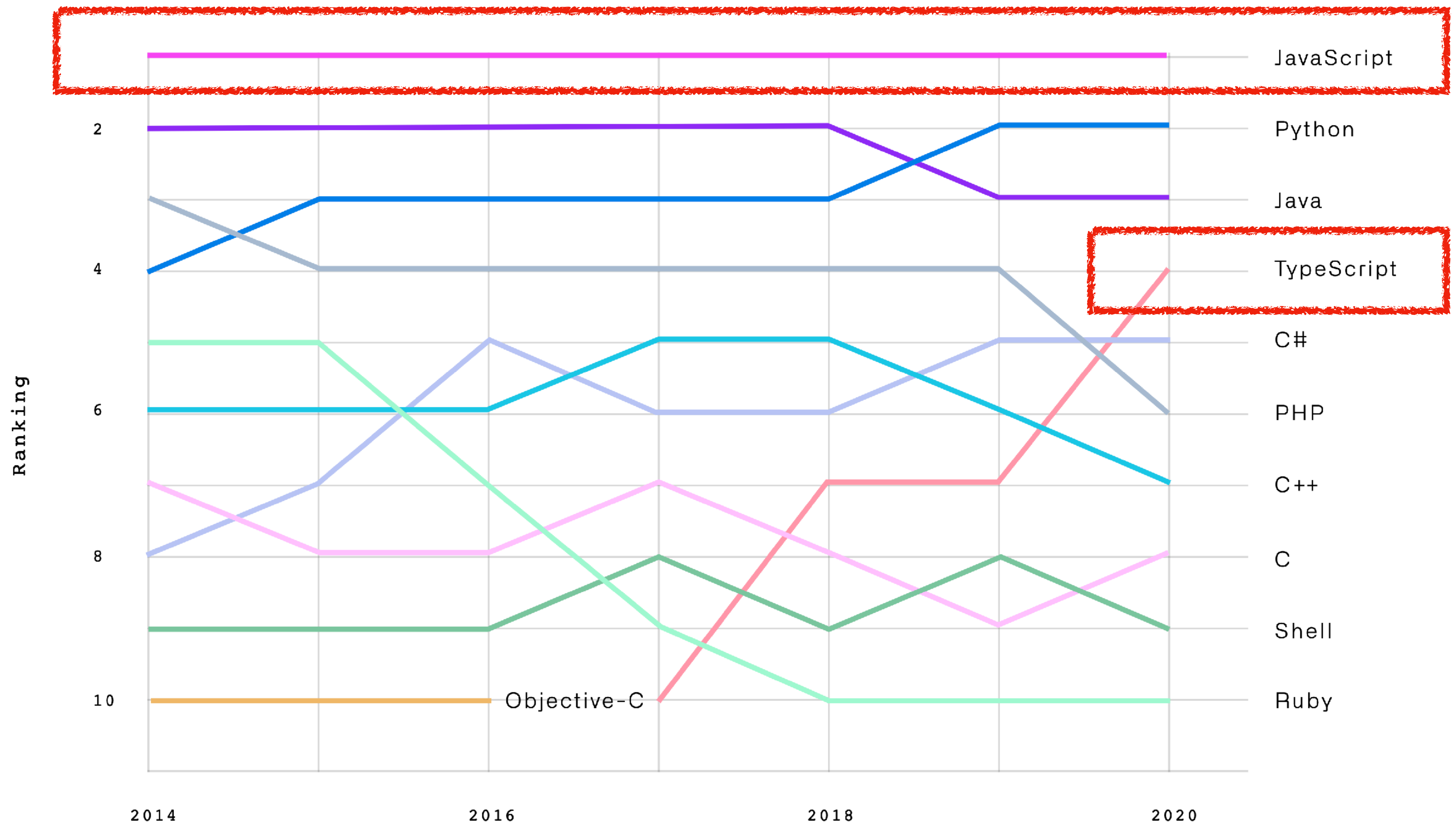
2021 한국컴퓨터종합학술대회 (KCC2021) Top Conference 세션

June 23, 2021

JavaScript is Everywhere



JavaScript is Dominating



https://octoverse.github.com/

JavaScript Complex Semantics

```
function f(x) { return x == !x; }
```

Always return `false`?

JavaScript Complex Semantics

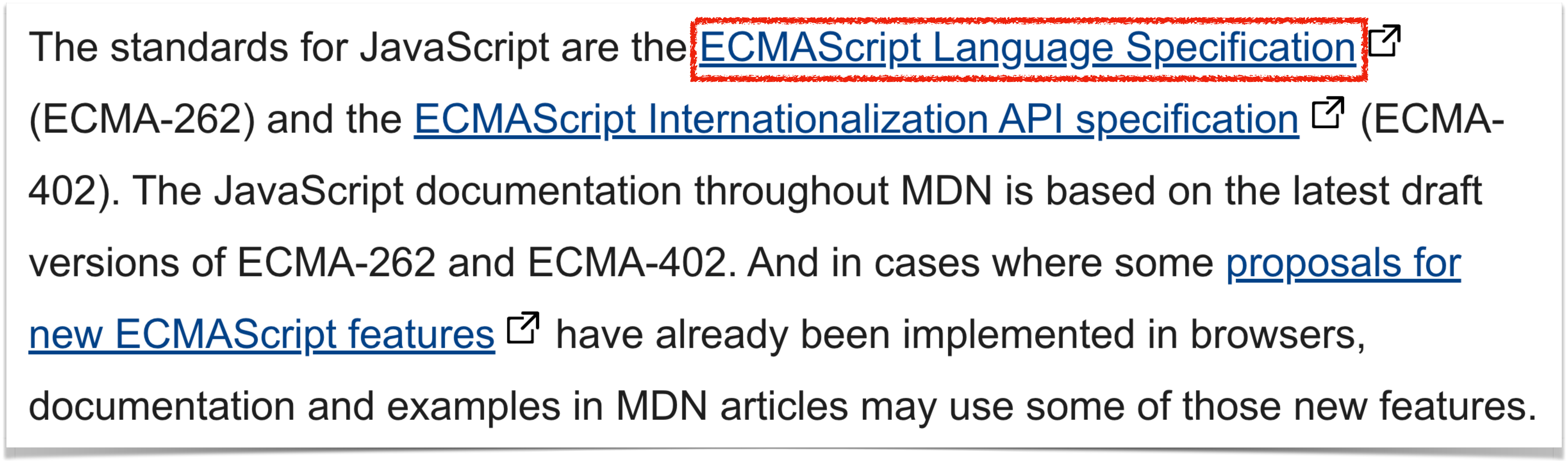
```
function f(x) { return x == !x; }
```

Always return `false`?

NO!!

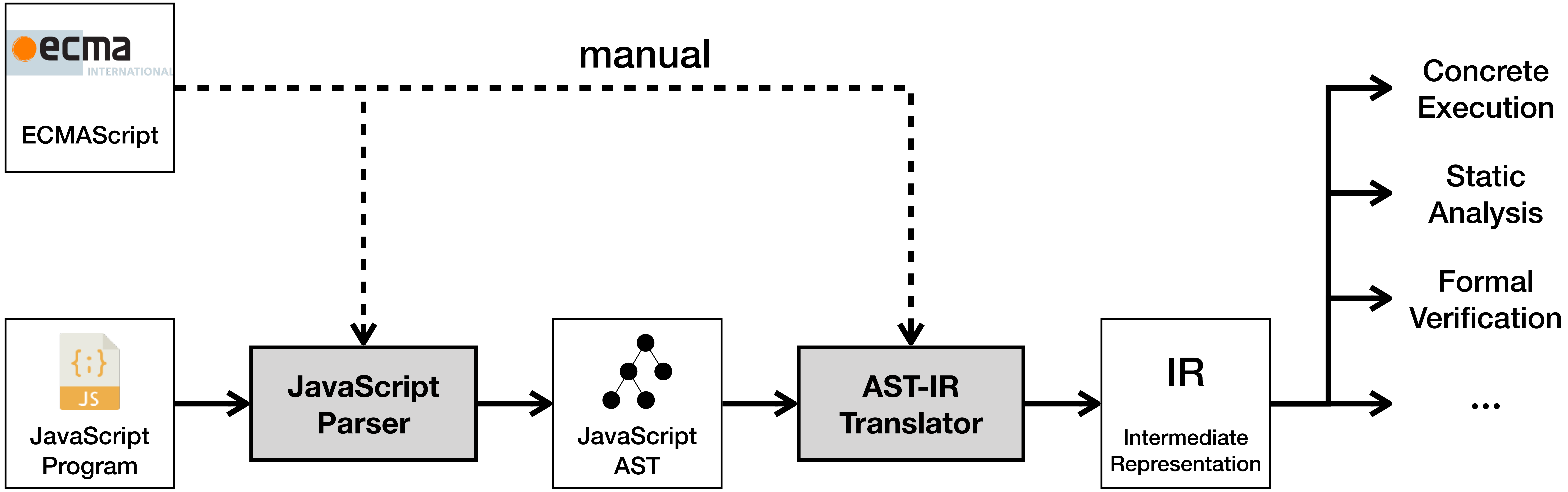
```
f([]) -> [] == ![]
      -> [] == false
      -> +[] == +false
      -> 0 == 0
      -> true
```

ECMAScript: JavaScript Specification

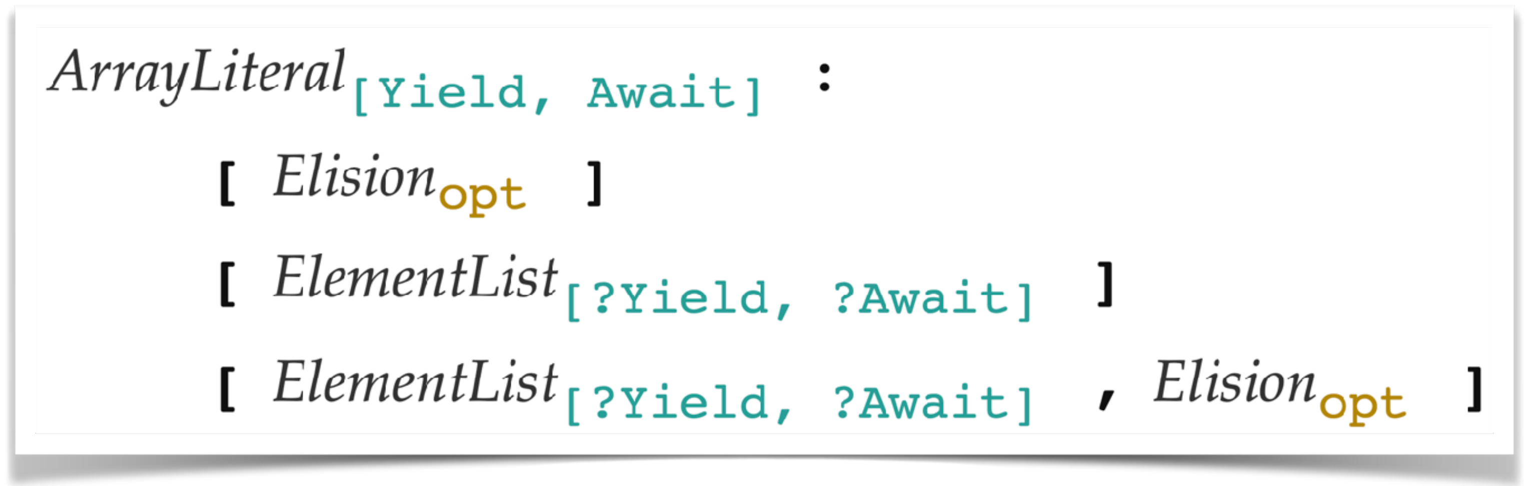

The standards for JavaScript are the ECMAScript Language Specification (ECMA-262) and the ECMAScript Internationalization API specification (ECMA-402). The JavaScript documentation throughout MDN is based on the latest draft versions of ECMA-262 and ECMA-402. And in cases where some proposals for new ECMAScript features have already been implemented in browsers, documentation and examples in MDN articles may use some of those new features.

https://developer.mozilla.org/en-US/docs/Web/JavaScript

IR-based Semantics Extraction

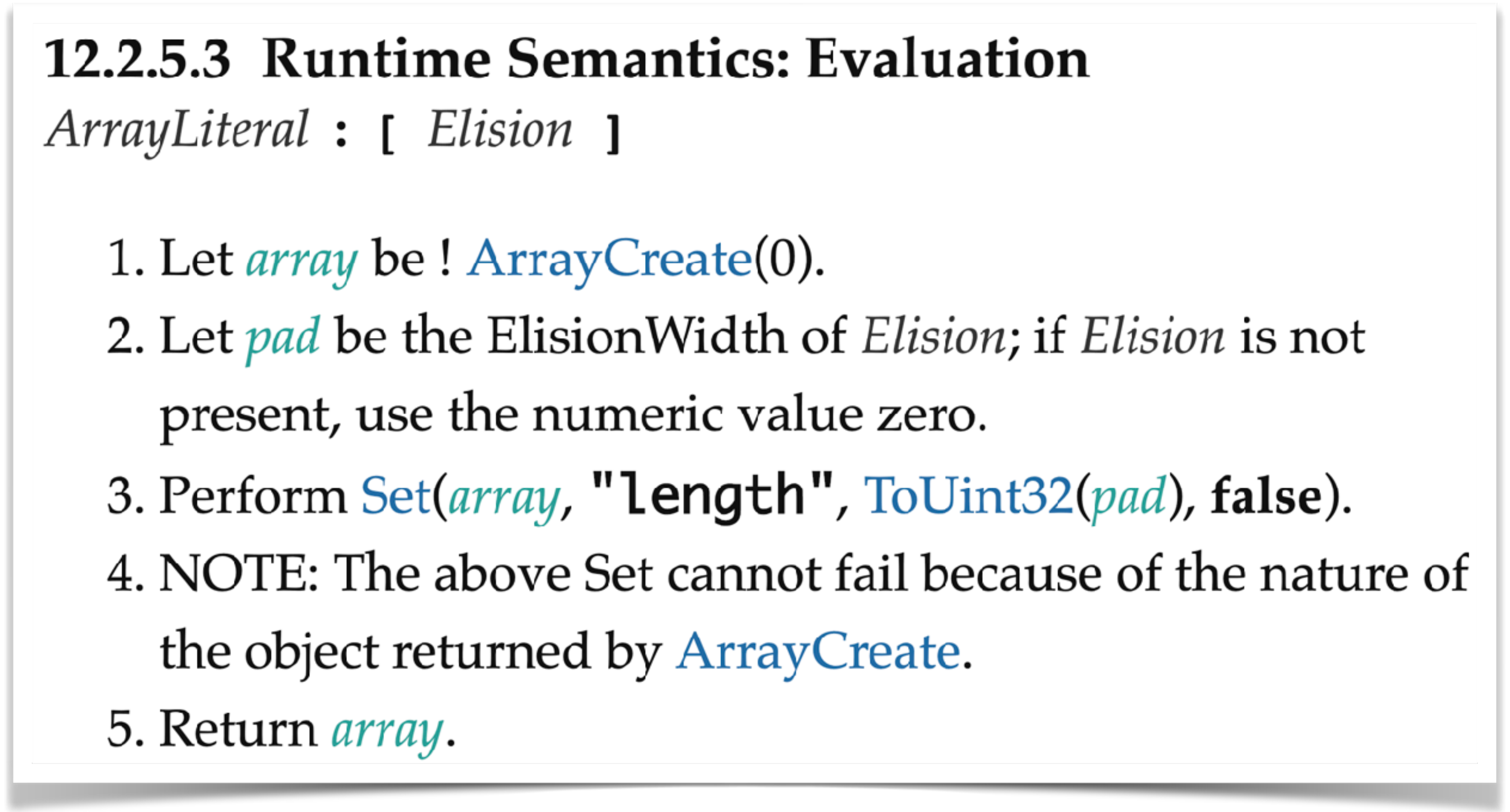


IR-based Semantics Extraction



ArrayLiteral`[Yield, Await]` :
  `[` *Elision*`opt` `]`
  `[` *ElementList*`[?Yield, ?Await]` `]`
  `[` *ElementList*`[?Yield, ?Await]` `,` *Elision*`opt` `]`

The production of *ArrayLiteral* in ES10

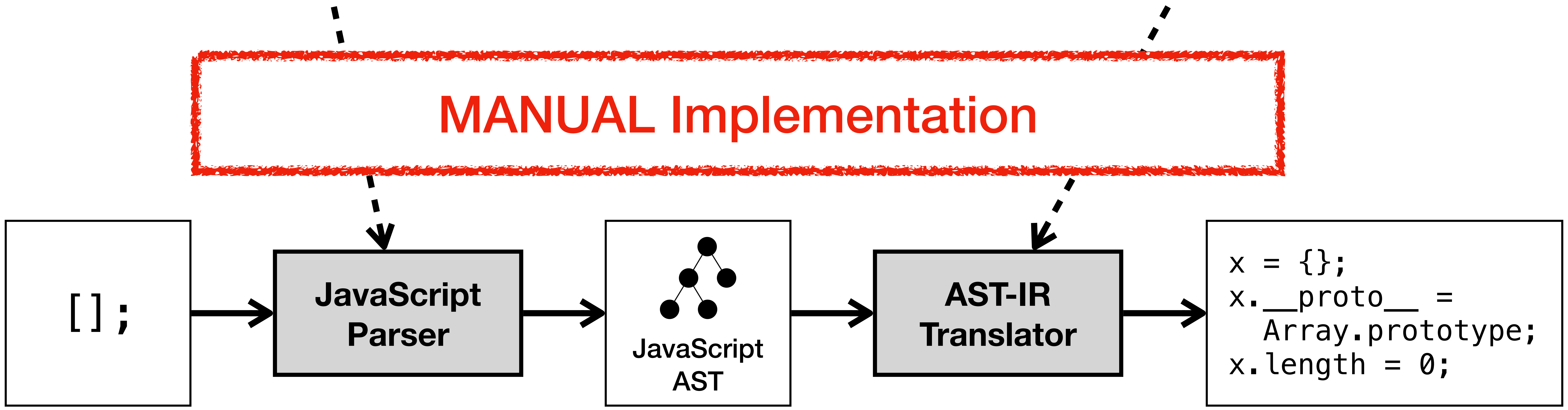


12.2.5.3 Runtime Semantics: Evaluation

ArrayLiteral : `[` *Elision* `]`

1. Let *array* be ! ArrayCreate(0).
2. Let *pad* be the ElisionWidth of *Elision*; if *Elision* is not present, use the numeric value zero.
3. Perform Set(*array*, `"length"`, ToUint32(*pad*), **false**).
4. NOTE: The above Set cannot fail because of the nature of the object returned by ArrayCreate.
5. Return *array*.

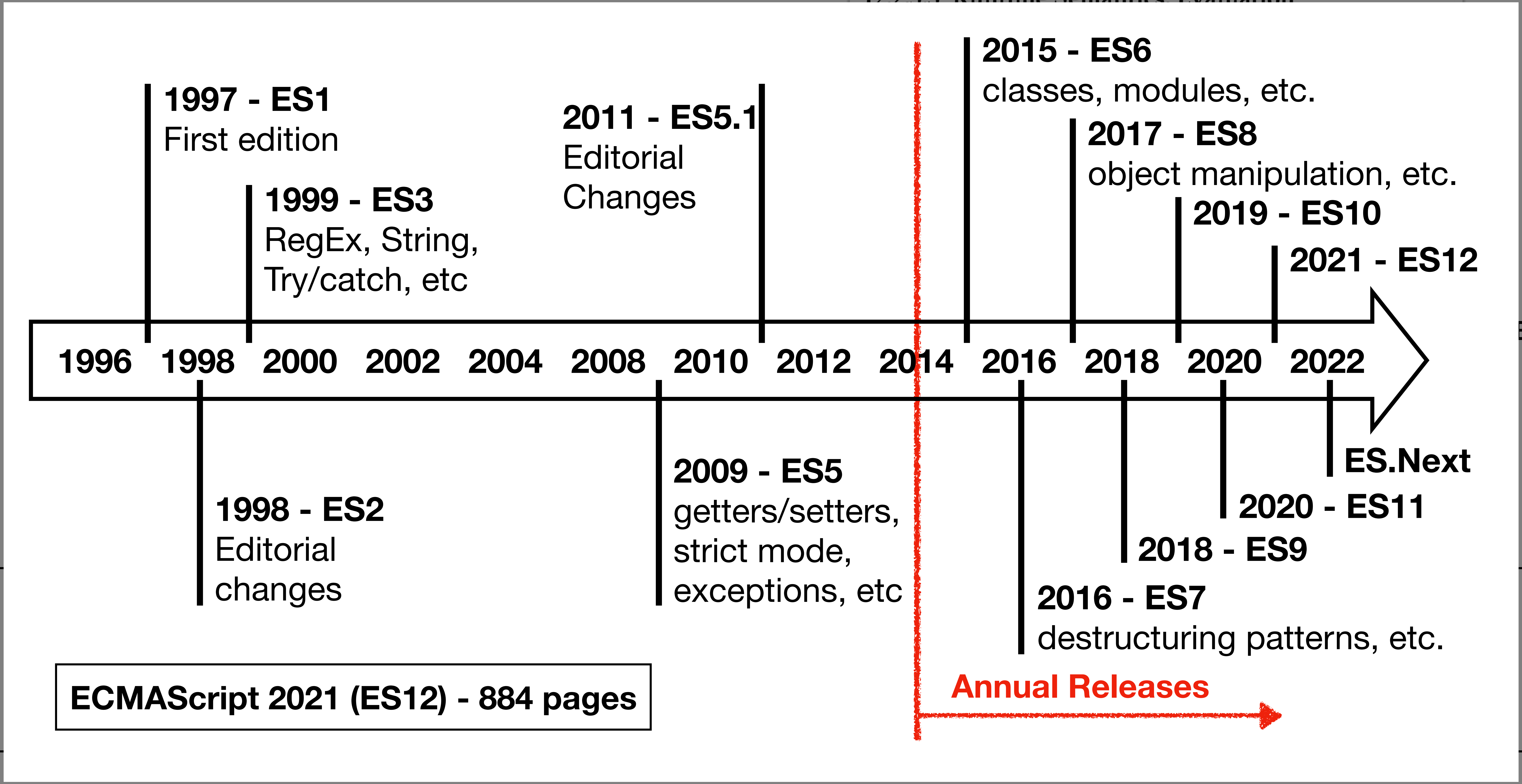
The Evaluation algorithm for the first alternative of *ArrayLiteral* in ES10



```
x = {};
x.__proto__ =
  Array.prototype;
x.length = 0;
```

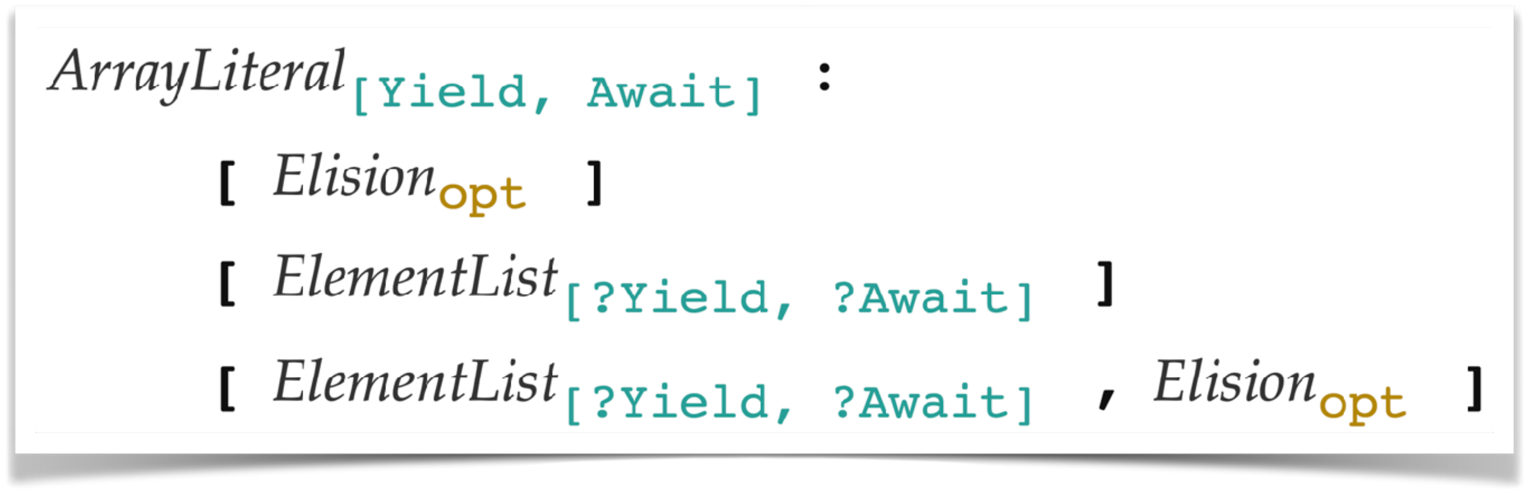
IR-based Semantics Extraction

12.2.5.3 Runtime Semantics: Evaluation



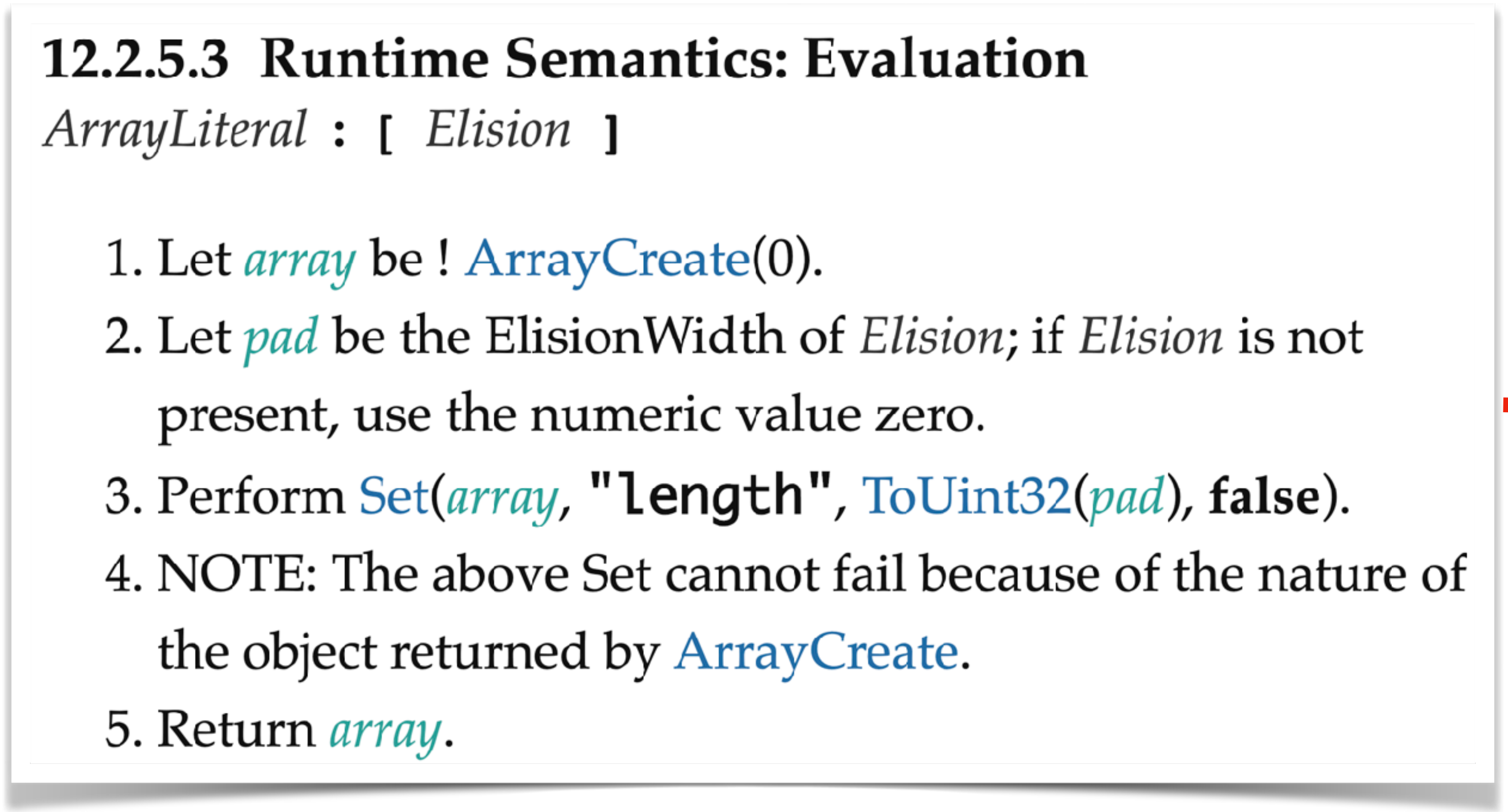
ES10

Our Approach



ArrayLiteral`[Yield, Await]` :
  `[` *Elision*`opt` `]`
  `[` *ElementList*`[?Yield, ?Await]` `]`
  `[` *ElementList*`[?Yield, ?Await]` `,` *Elision*`opt` `]`

The production of *ArrayLiteral* in ES10

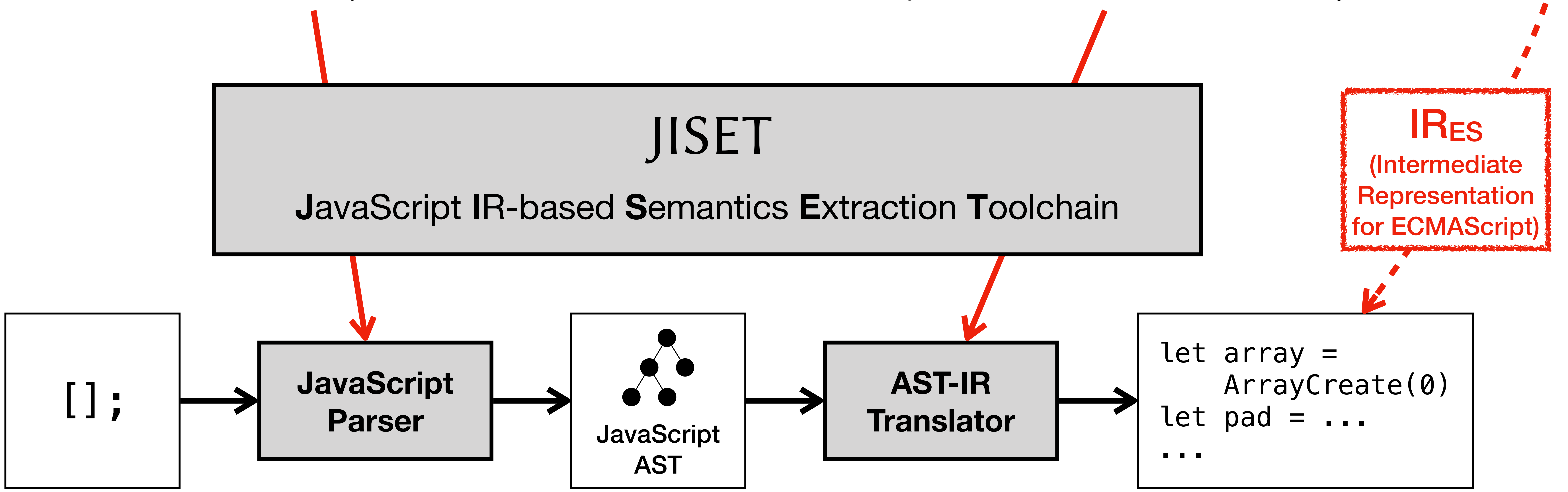


12.2.5.3 Runtime Semantics: Evaluation

ArrayLiteral : `[` *Elision* `]`

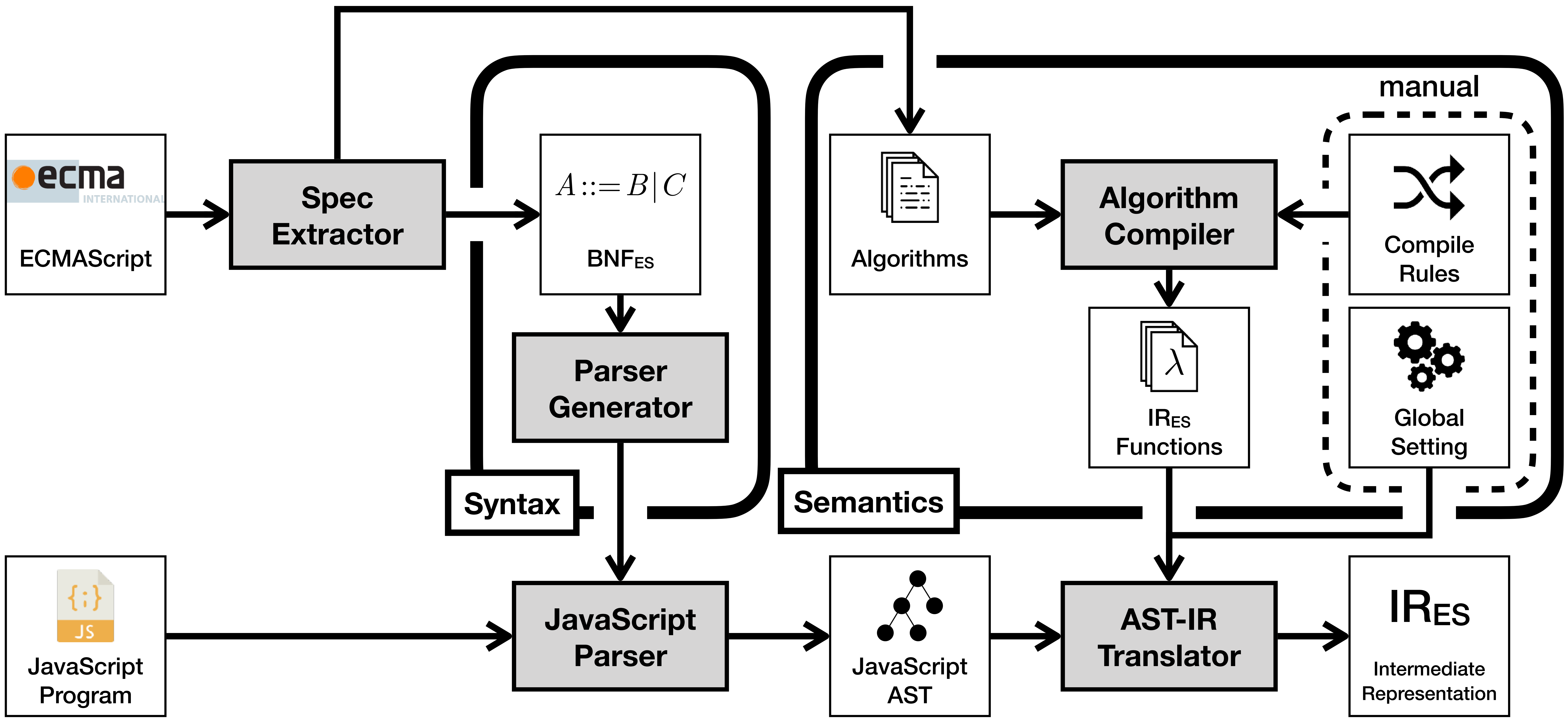
1. Let *array* be ! ArrayCreate(0).
2. Let *pad* be the ElisionWidth of *Elision*; if *Elision* is not present, use the numeric value zero.
3. Perform Set(*array*, `"length"`, ToUint32(*pad*), **false**).
4. NOTE: The above Set cannot fail because of the nature of the object returned by ArrayCreate.
5. Return *array*.

The Evaluation **algorithm for the first alternative of *ArrayLiteral* in ES10**

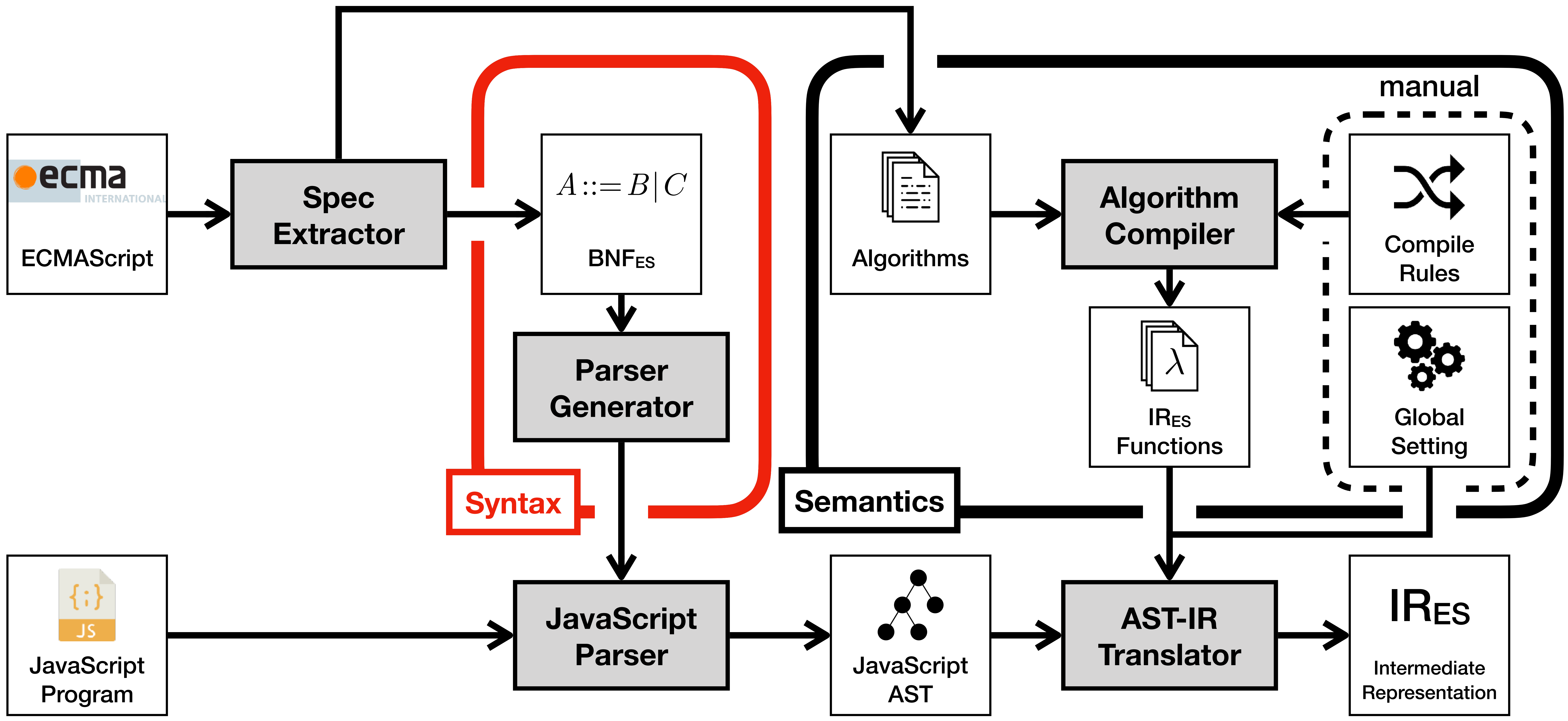

JISET
JavaScript IR-based Semantics Extraction Toolchain
IR_ES
(Intermediate Representation for ECMAScript)
[];
JavaScript Parser
JavaScript AST
AST-IR Translator

```
let array =
    ArrayCreate(0)
let pad = ...
...
```

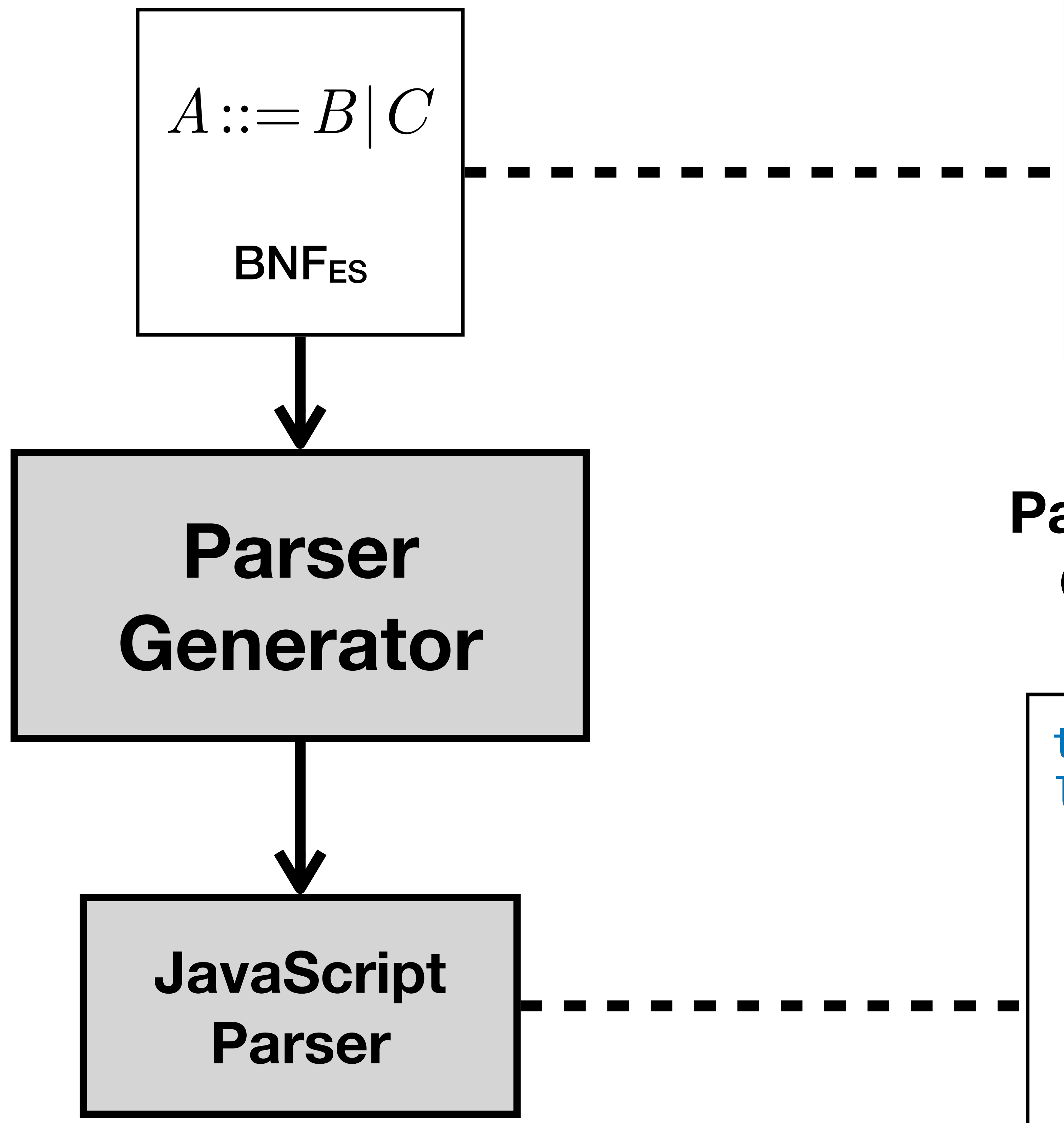
Overall Structure of JISET

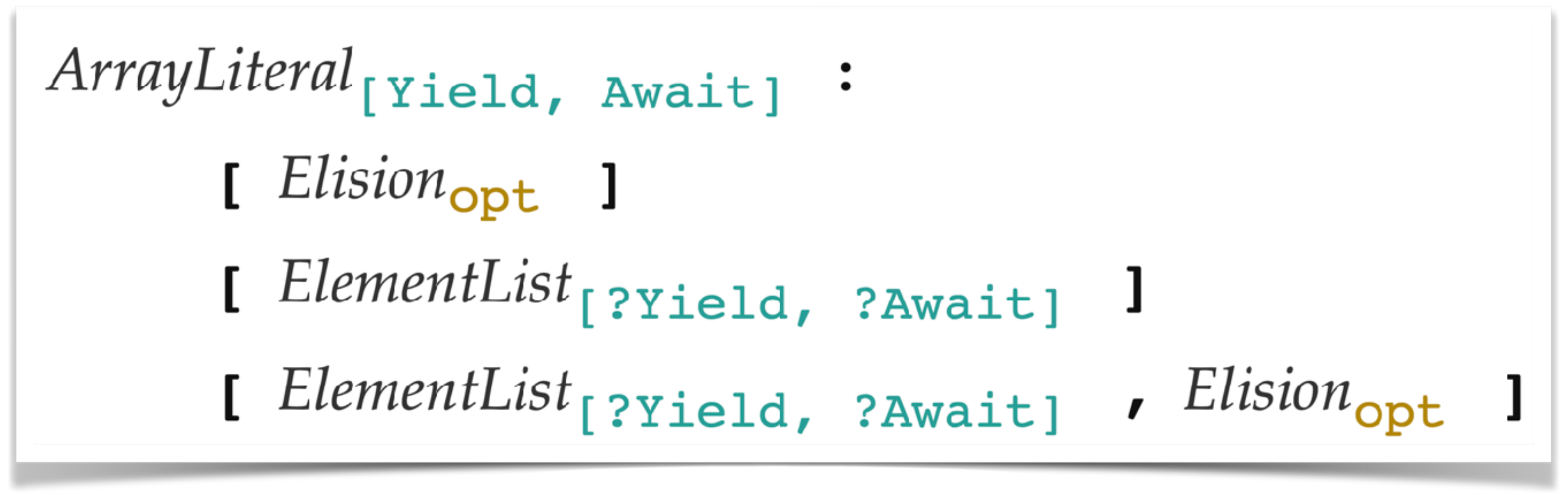


Overall Structure of JISET



Syntax - Parser Synthesis





ArrayLiteral`[Yield, Await]` **:**

  [*Elision*`opt` **]**

  [*ElementList*`[?Yield, ?Await]` **]**

  [*ElementList*`[?Yield, ?Await]` **,** *Elision*`opt` **]**

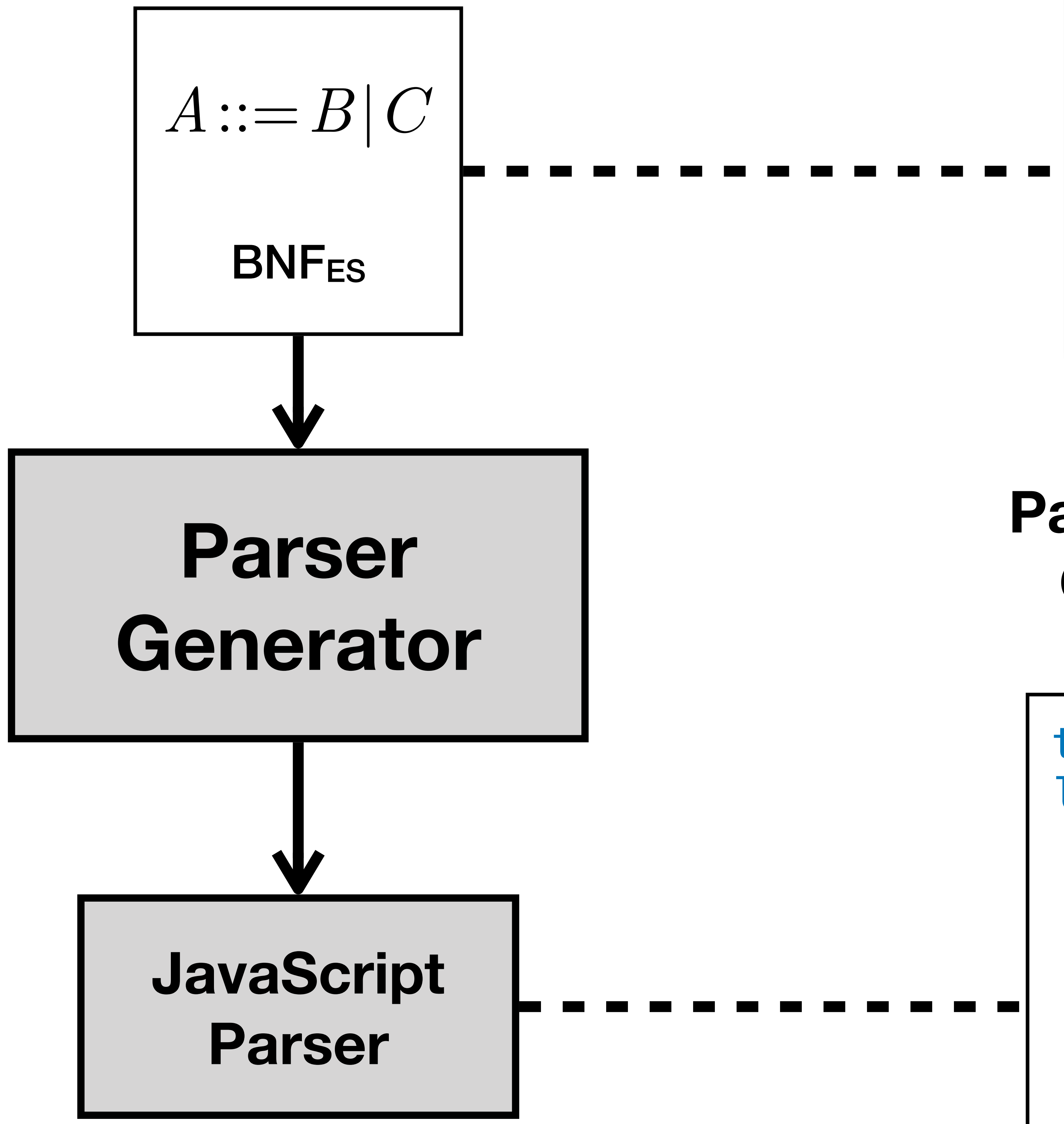
The production of *ArrayLiteral* in ES10

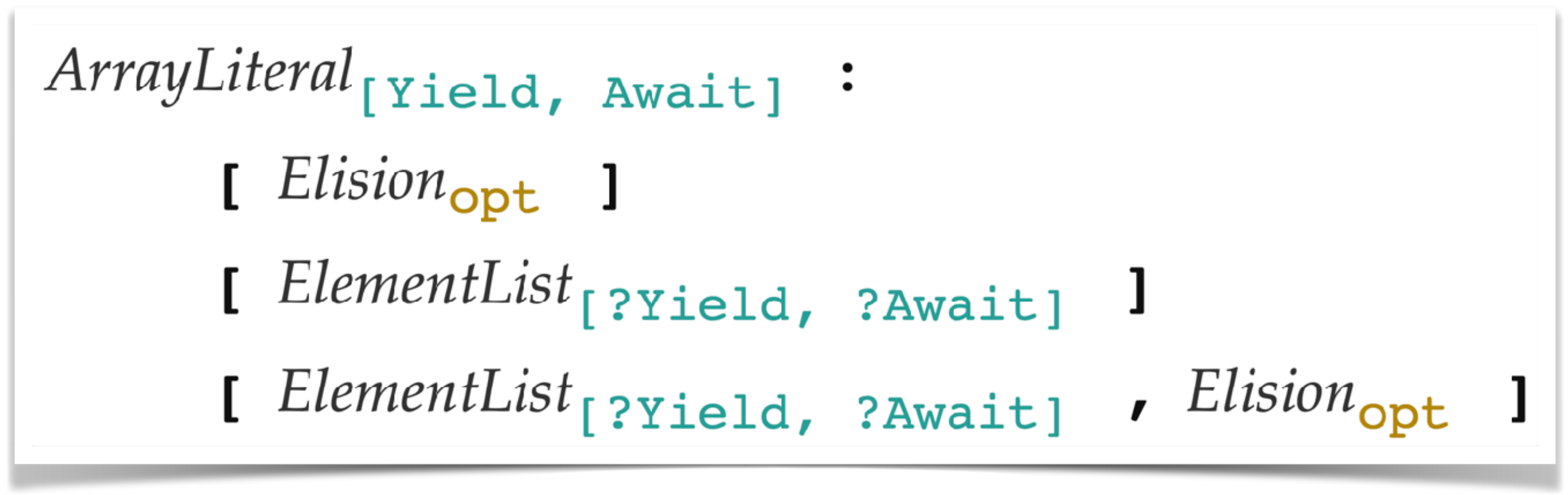
Parsing Expression Grammar (PEG)

```
type P[T] = List[Boolean] => LAParser[T]
lazy val ArrayLiteral: P[ArrayLiteral] = memo {
  case List(Yield, Await) =>
  "[" ~ opt(Elision) ~ "]"
      ^^ ArrayLiteral0 |
  "[" ~ ElementList(Yield, Await) ~ "]"
      ^^ ArrayLiteral1 |
  "[" ~ ElementList(Yield, Await)
      ~ "," ~ opt(Elision) ~ "]"
      ^^ ArrayLiteral2
}
```

The generated parser for *ArrayLiteral* in ES10

Syntax - Parser Synthesis

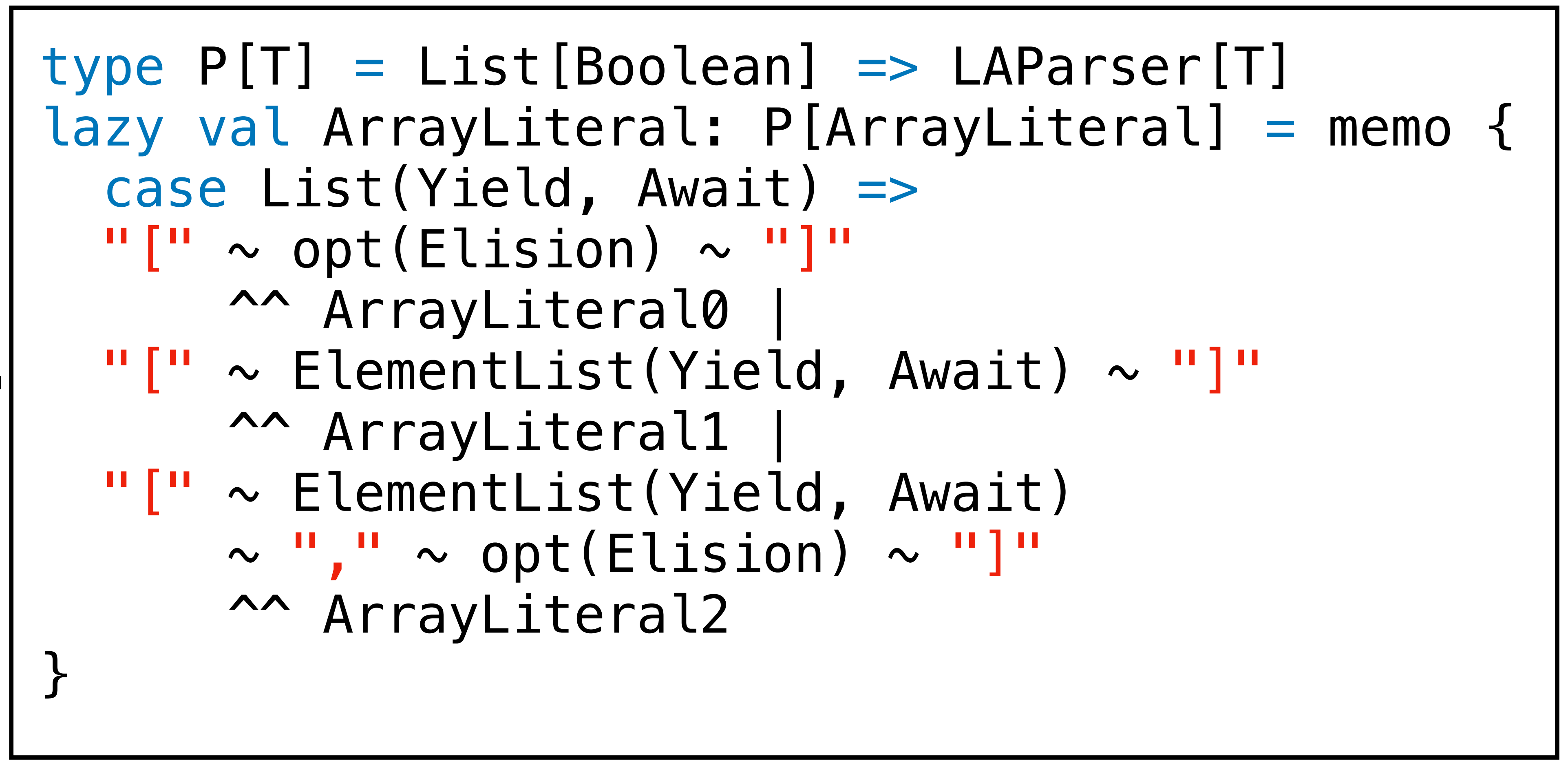




ArrayLiteral`[Yield, Await]` :
  `[` *Elision*`opt` `]`
  `[` *ElementList*`[?Yield, ?Await]` `]`
  `[` *ElementList*`[?Yield, ?Await]` `,` *Elision*`opt` `]`

The production of *ArrayLiteral* in ES10

Parsing Expression Grammar (PEG) + **Lookahead Parsing**



```
type P[T] = List[Boolean] => LAParser[T]
lazy val ArrayLiteral: P[ArrayLiteral] = memo {
  case List(Yield, Await) =>
  "[" ~ opt(Elision) ~ "]"
      ^^ ArrayLiteral0 |
  "[" ~ ElementList(Yield, Await) ~ "]"
      ^^ ArrayLiteral1 |
  "[" ~ ElementList(Yield, Await)
      ~ "," ~ opt(Elision) ~ "]"
      ^^ ArrayLiteral2
}
```

The generated parser for *ArrayLiteral* in ES10

Syntax - Lookahead Parsing

$$
\begin{aligned}
\mathbf{first}_\alpha(s_1 \cdots s_n) &= \mathbf{first}_s(s_1) :+ \mathbf{first}_s(s_2 \cdots s_n) \\
&\quad \text{where } x :+ y = \begin{cases} x \cup y & \text{if } \circ \in x \\ x & \text{otherwise} \end{cases} \\
\mathbf{first}_s(\epsilon) &= \{\circ\} \\
\mathbf{first}_s(\mathsf{a}) &= \{\mathsf{a}\} \\
\mathbf{first}_s(A(a_1, \cdots, a_k)) &= \mathbf{first}_\alpha(\alpha_1) \cup \cdots \cup \mathbf{first}_\alpha(\alpha_n) \\
&\quad \text{where } A(a_1, \cdots, a_k) = \alpha_1 \mid \cdots \mid \alpha_n \\
\mathbf{first}_s(s?) &= \mathbf{first}_s(s) \cup \{\circ\} \\
\mathbf{first}_s(+s) &= \mathbf{first}_s(s) \\
\mathbf{first}_s(-s) &= \{\circ\} \\
\mathbf{first}_s(s \setminus s') &= \mathbf{first}_s(s) \\
\mathbf{first}_s(\langle \neg \mathsf{LT} \rangle) &= \{\circ\}
\end{aligned}
$$

Algorithm for first tokens of BNF$_{\mathbf{ES}}$

Algorithm for lookahead parsing

$$
\begin{aligned}
(s_1 \cdots s_n)[L] &= s_1[\mathbf{first}_s(s_2 \cdots s_n) :+ L]\ (s_1 \cdots s_n)[L] \\
\epsilon[L] &= +\mathbf{get}_s(L) \\
\mathsf{a}[L] &= \mathsf{a}\ + \mathbf{get}_s(L) \\
A(a_1, \cdots, a_k)[L] &= \alpha_1[L] \mid \cdots \mid \alpha_n[L] \\
&\quad \text{where } A(a_1, \cdots, a_k) = \alpha_1 \mid \cdots \mid \alpha_n \\
s?[L] &= s[L] \mid \epsilon[L] \\
(\pm s)[L] &= \pm(s[L]) \\
(s \setminus s')[L] &= s[L] \setminus s' \\
\langle \neg \mathsf{LT} \rangle &= \langle \neg \mathsf{LT} \rangle\ + \mathbf{get}_s(L)
\end{aligned}
$$

Syntax - Evaluation

Version	ES7	ES8	ES9	ES10	Average
# Lexical productions	78	78	78	81	78.75
# Syntactic productions	157	167	167	174	166.25

Old version	ES7	ES8	ES9	Average
New version	ES8	ES9	ES10	
Δ # Lexical productions	3	5	6	4.67
Δ # Syntactic productions	140	15	8	54.33

Syntax - Evaluation

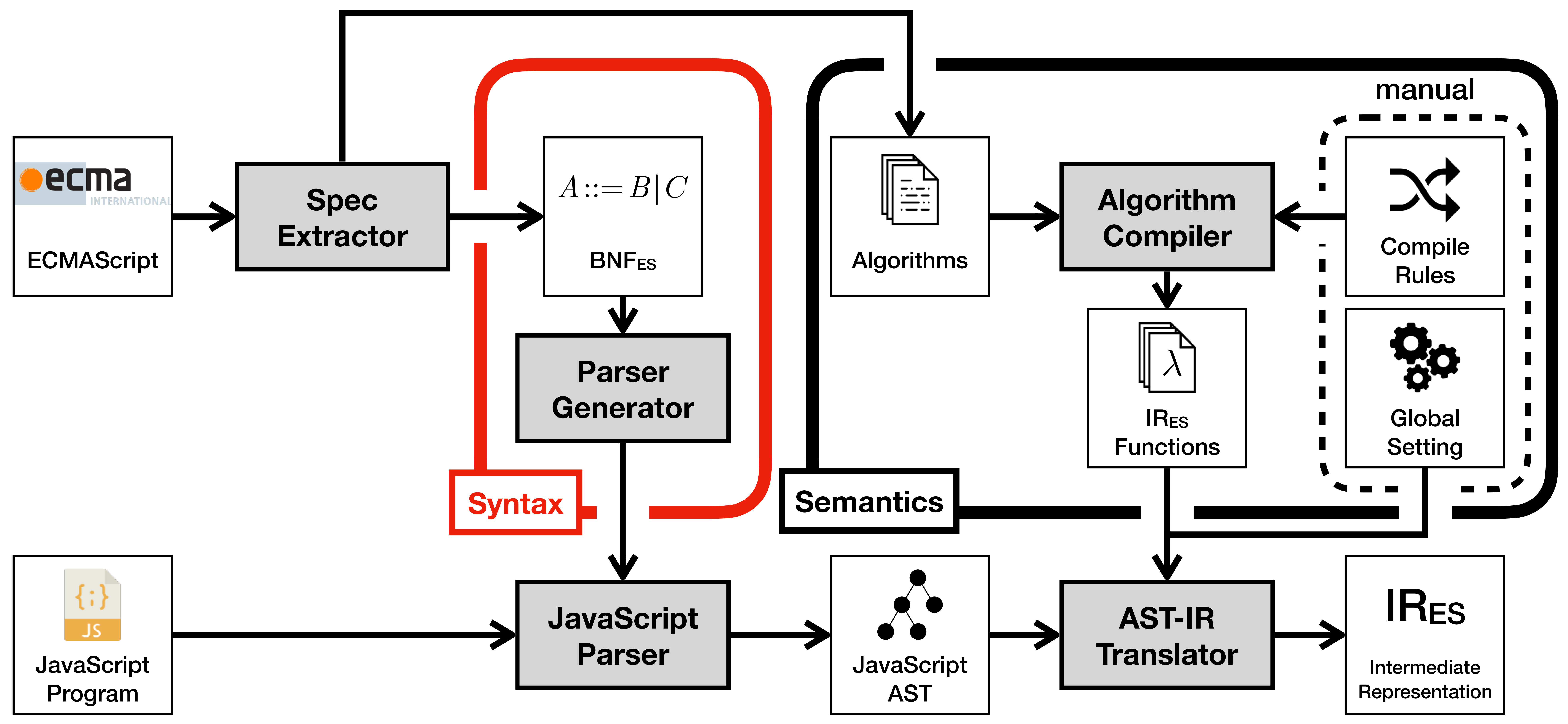
All Success!!

Version	ES7	ES8	ES9	ES10	Average
# Lexical productions	78	78	78	81	78.75
# Syntactic productions	157	167	167	174	166.25

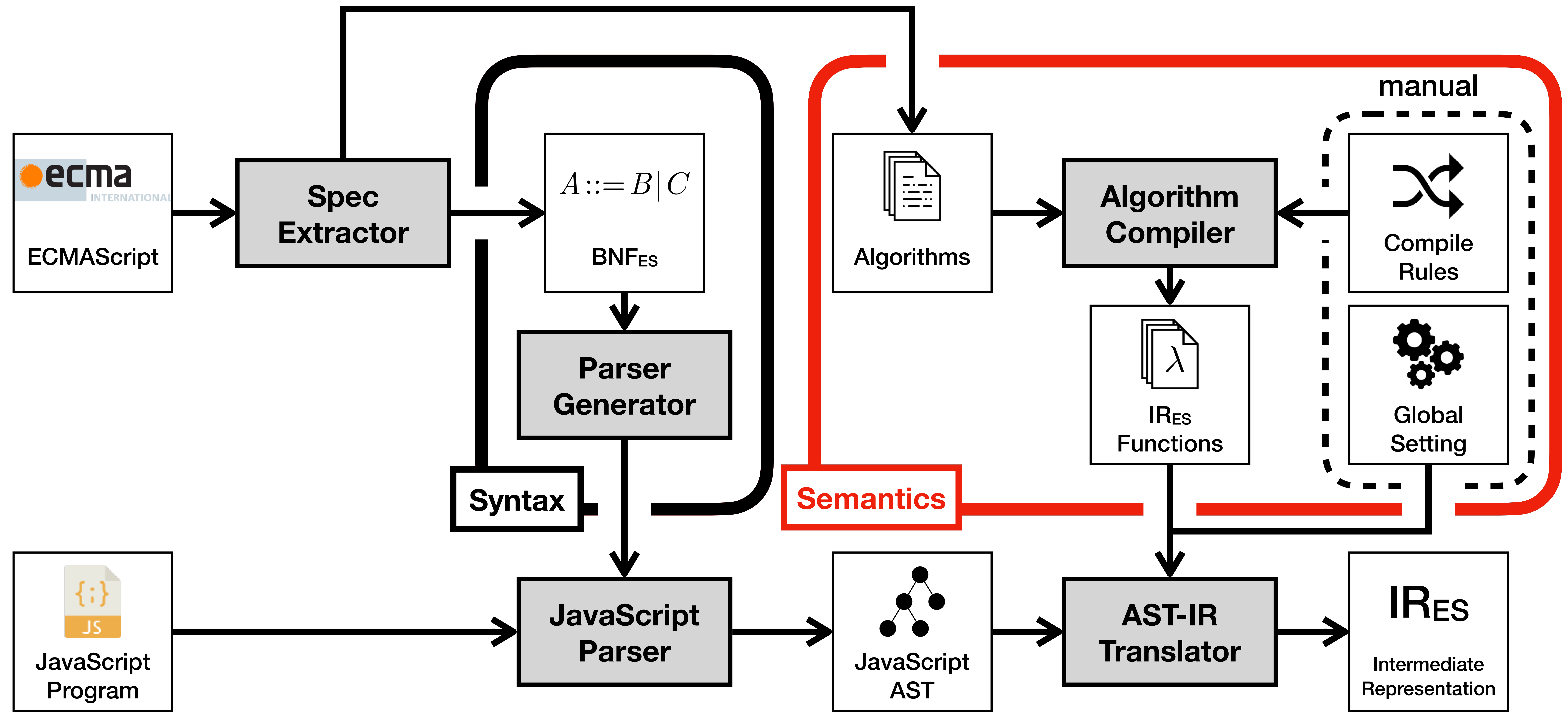
Pass all parsing tests
in Test262

Old version	ES7	ES8	ES9	Average
New version	ES8	ES9	ES10	
Δ # Lexical productions	3	5	6	4.67
Δ # Syntactic productions	140	15	8	54.33

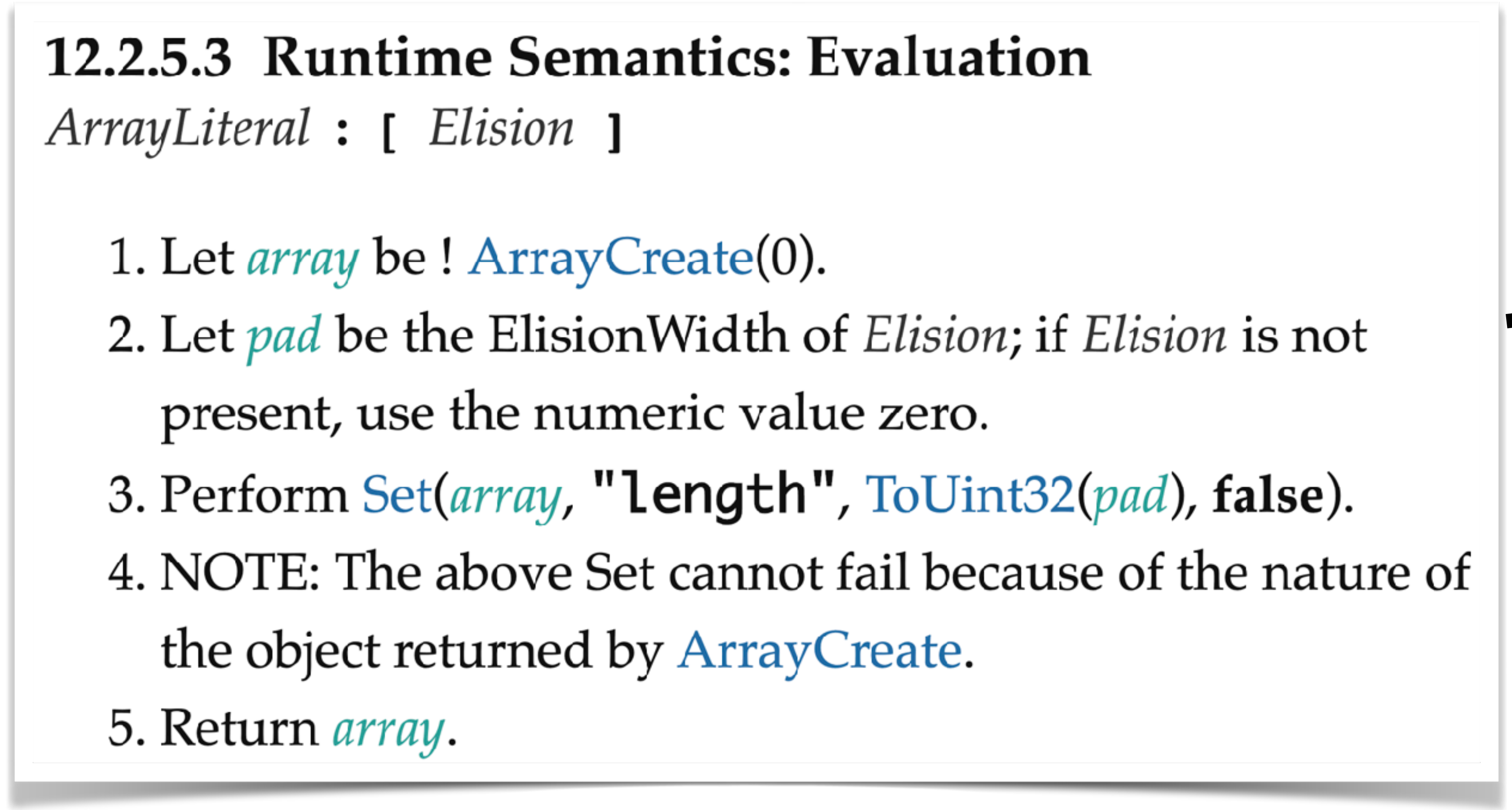
Overall Structure of JISET



Overall Structure of JISET



Semantics - Algorithm Compilation



12.2.5.3 Runtime Semantics: Evaluation

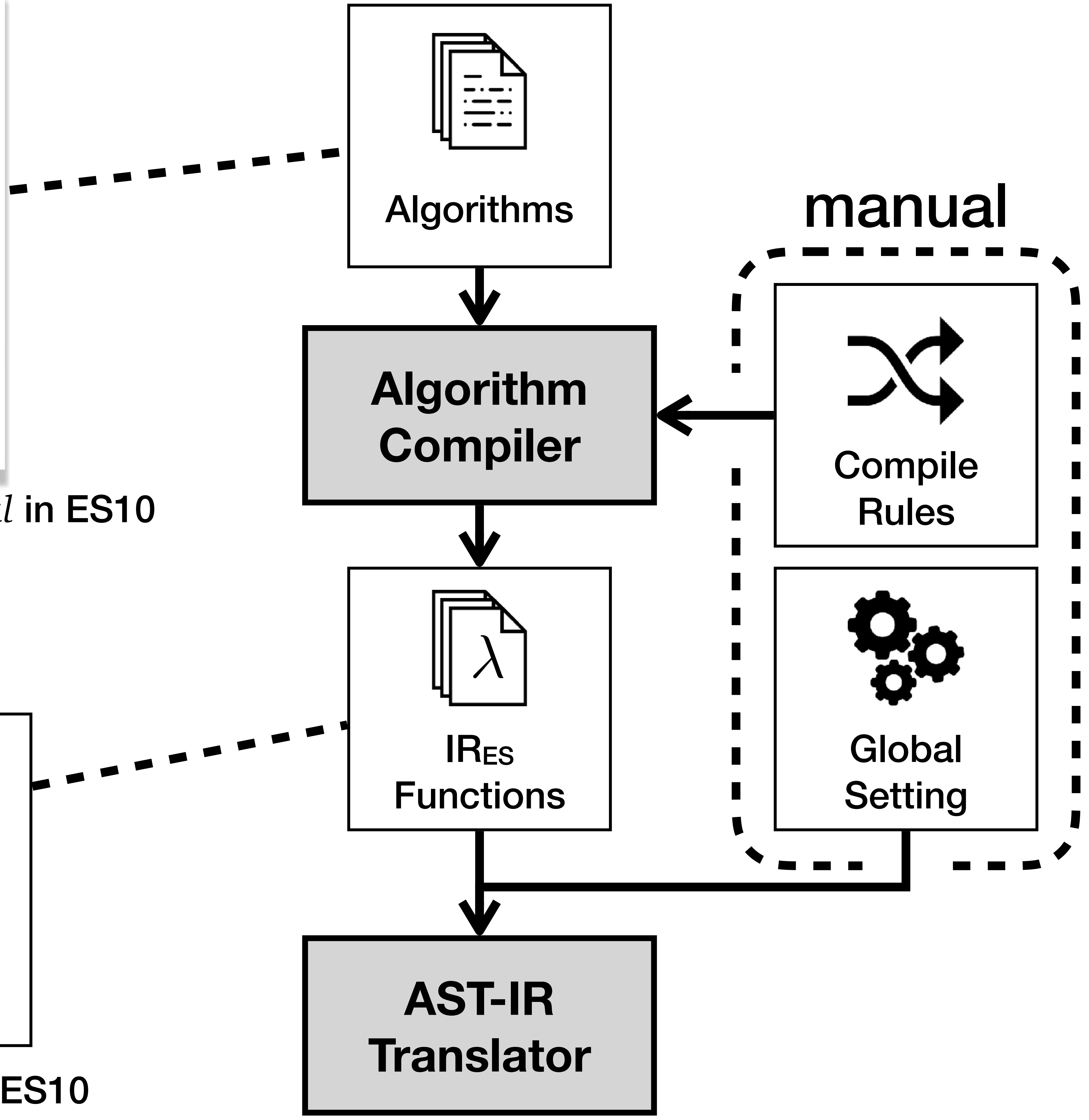
ArrayLiteral : `[` *Elision* `]`

1. Let *array* be ! ArrayCreate(0).
2. Let *pad* be the ElisionWidth of *Elision*; if *Elision* is not present, use the numeric value zero.
3. Perform Set(*array*, `"length"`, ToUint32(*pad*), **false**).
4. NOTE: The above Set cannot fail because of the nature of the object returned by ArrayCreate.
5. Return *array*.

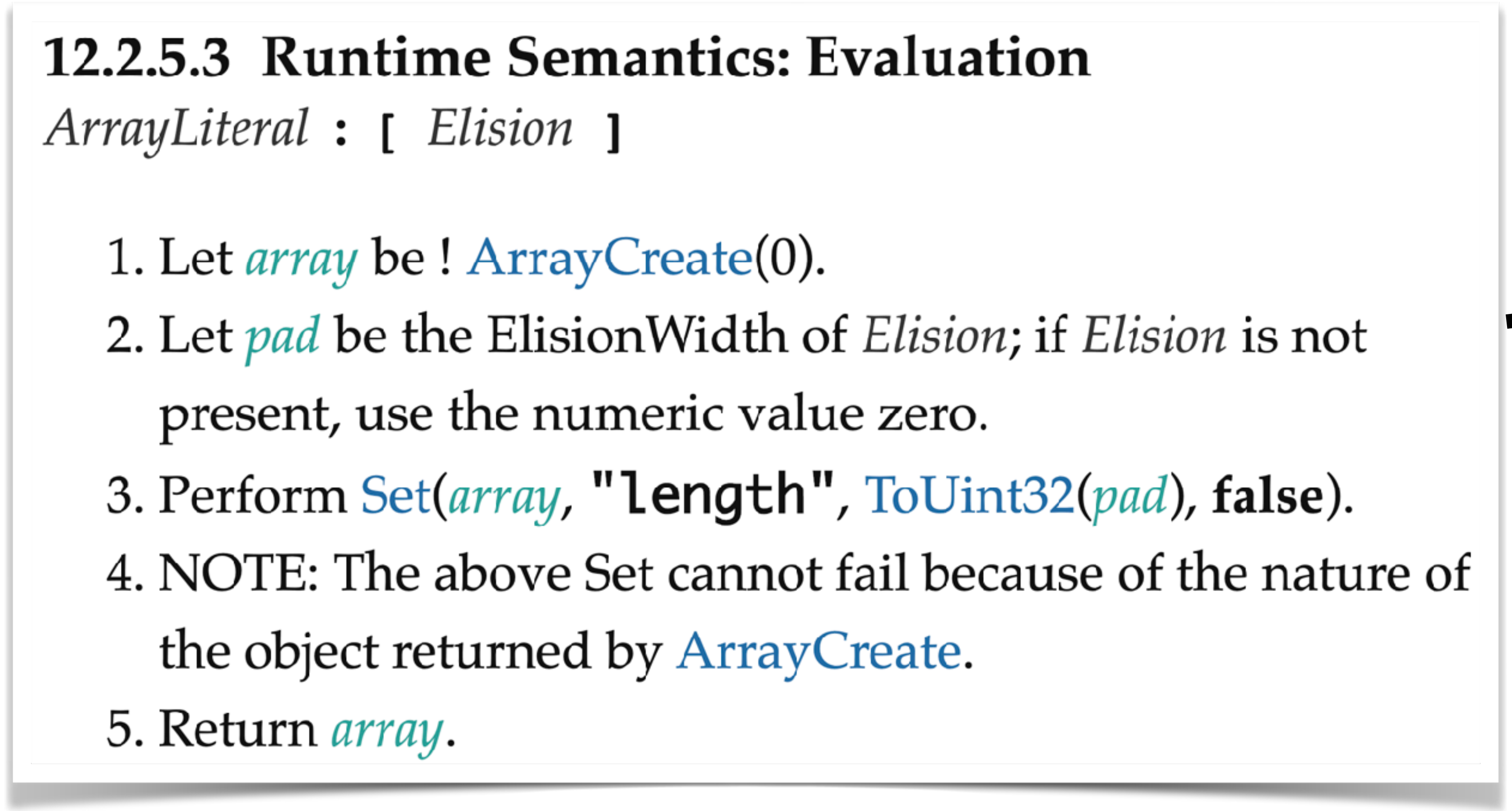
The Evaluation **algorithm for the first alternative of** *ArrayLiteral* **in ES10**

```
"ArrayLiteral0.Evaluation" (Elision) => {
  let array = ! (ArrayCreate 0)
  if (= Elision absent) let pad = 0
  else let pad = Elision.ElisionWidth
  (Set array "length" (ToUint32 pad) false)
  return array
}
```

The IR$_{\text{ES}}$ **function of the first alternative of** *ArrayLiteral* **in ES10**



Semantics - Algorithm Compilation



12.2.5.3 Runtime Semantics: Evaluation

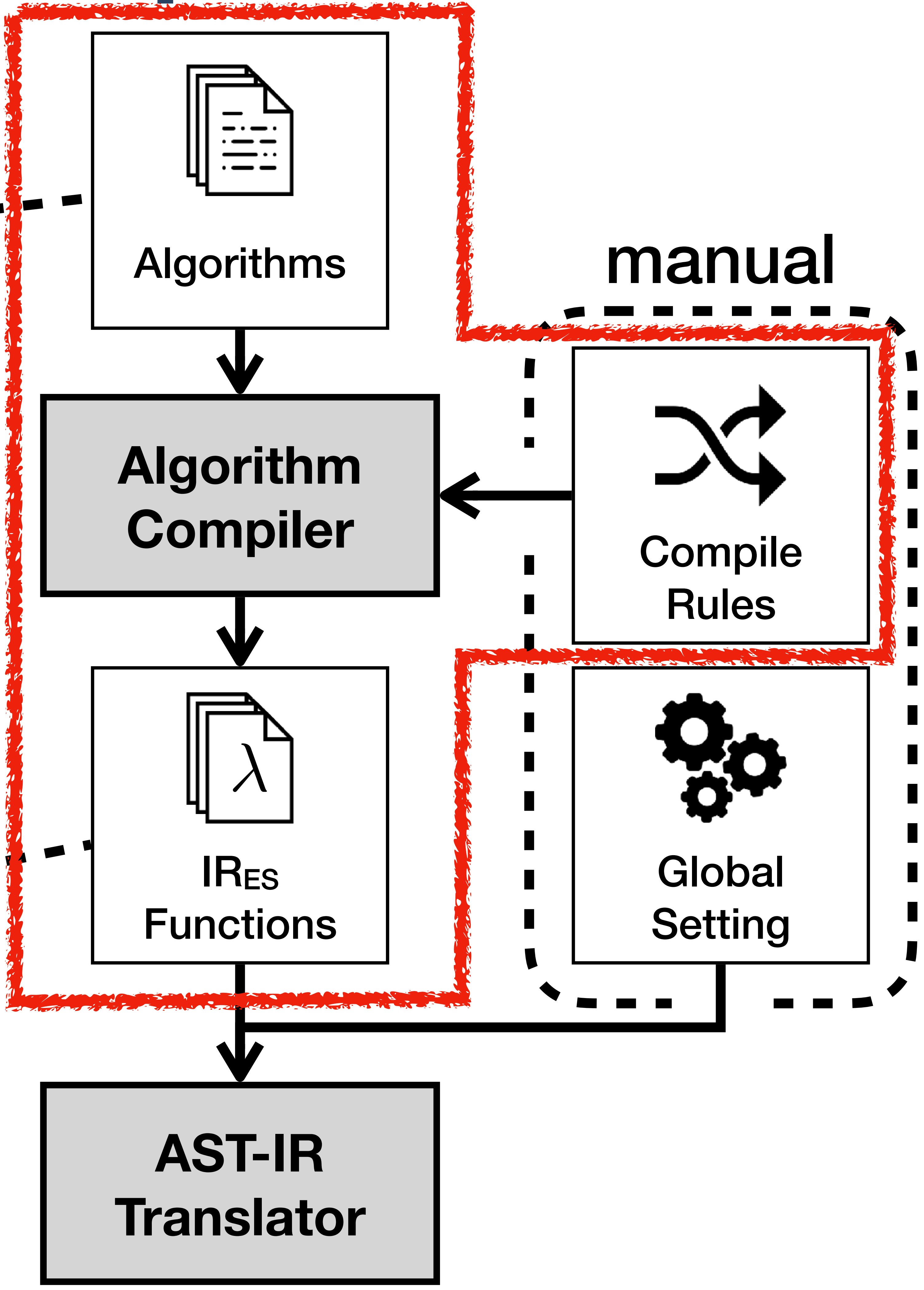
ArrayLiteral : [*Elision*]

1. Let *array* be ! ArrayCreate(0).
2. Let *pad* be the ElisionWidth of *Elision*; if *Elision* is not present, use the numeric value zero.
3. Perform Set(*array*, "length", ToUint32(*pad*), **false**).
4. NOTE: The above Set cannot fail because of the nature of the object returned by ArrayCreate.
5. Return *array*.

The Evaluation **algorithm for the first alternative of** *ArrayLiteral* **in ES10**

```
"ArrayLiteral0.Evaluation" (Elision) => {
  let array = ! (ArrayCreate 0)
  if (= Elision absent) let pad = 0
  else let pad = Elision.ElisionWidth
  (Set array "length" (ToUint32 pad) false)
  return array
}
```

The IR$_{ES}$ function of the first alternative of *ArrayLiteral* **in ES10**



Parsing rules   Conversion Rules

```
S = // statements
  Let ~ V ~ be ~ E ~ . ^^ ILet

E = // expressions
  ! E                  ^^ EAbruptCheck |
  str ~ ( ~ E ~ )      ^^ ECall        |
  num                  ^^ _.toDouble
```

Simplified compile rules

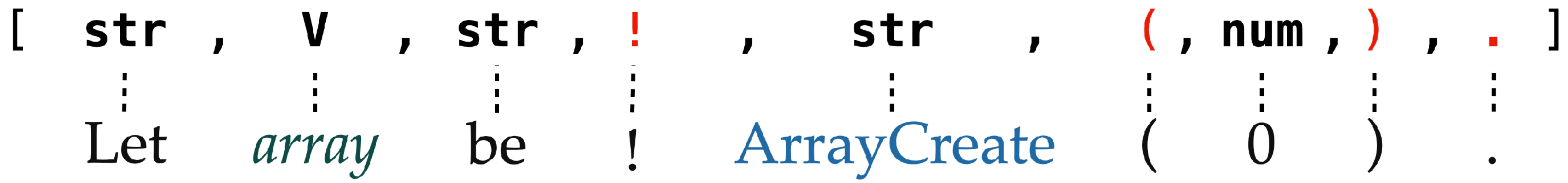
Let *array* be ! ArrayCreate (0) .

Parsing rules Conversion Rules

```
S = // statements
  Let ~ V ~ be ~ E ~ . ^^ ILet

E = // expressions
  ! E                 ^^ EAbruptCheck |
  str ~ ( ~ E ~ )     ^^ ECall        |
  num                 ^^ _.toDouble
```

Simplified compile rules



Parsing rules   Conversion Rules

```
S = // statements
  Let ~ V ~ be ~ E ~ .  ^^ ILet

E = // expressions
  ! E                   ^^ EAbruptCheck |
  str ~ ( ~ E ~ )       ^^ ECall        |
  num                   ^^ _.toDouble
```

Simplified compile rules

```
[ str ,   V   , str , ! ,     str     , ( , num , ) , . ]
  Let   array   be    !   ArrayCreate   (    0    )   .
```

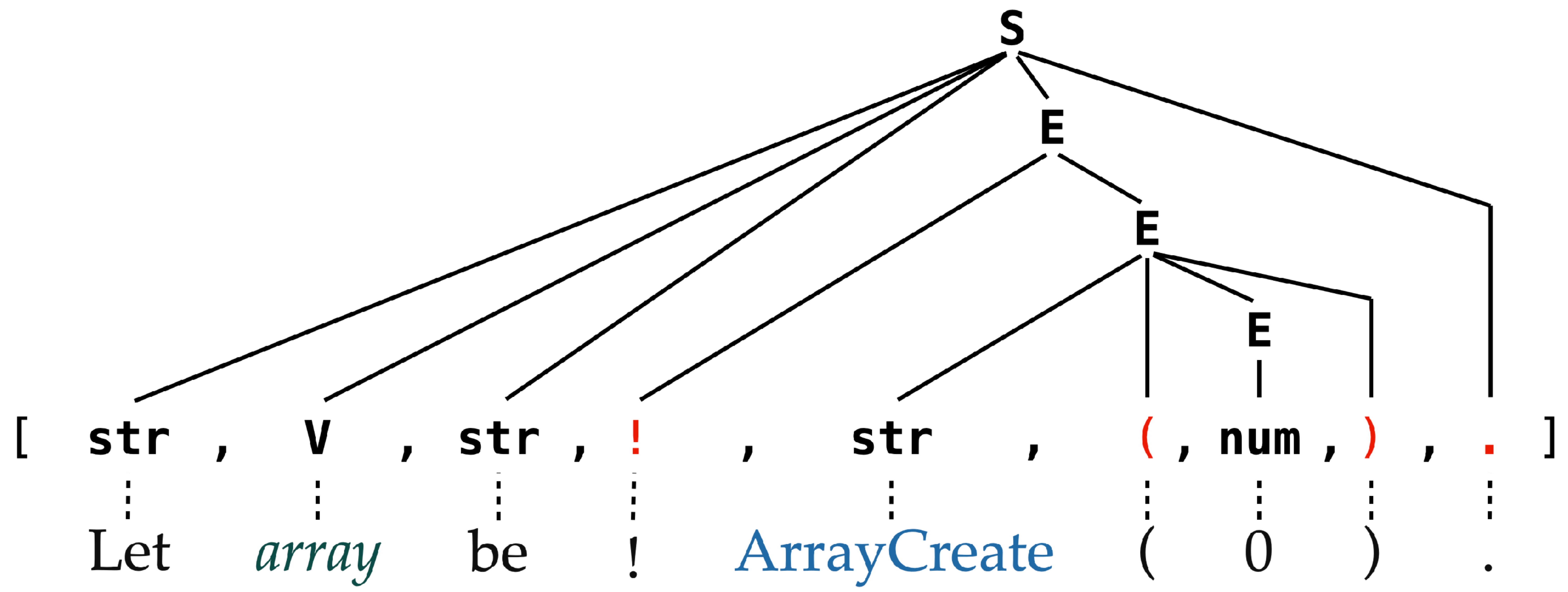
Parsing rules

Conversion Rules

```
S = // statements
  Let ~ V ~ be ~ E ~ .  ^^ ILet

E = // expressions
  ! E                   ^^ EAbruptCheck |
  str ~ ( ~ E ~ )       ^^ ECall        |
  num                   ^^ _.toDouble
```

Simplified compile rules

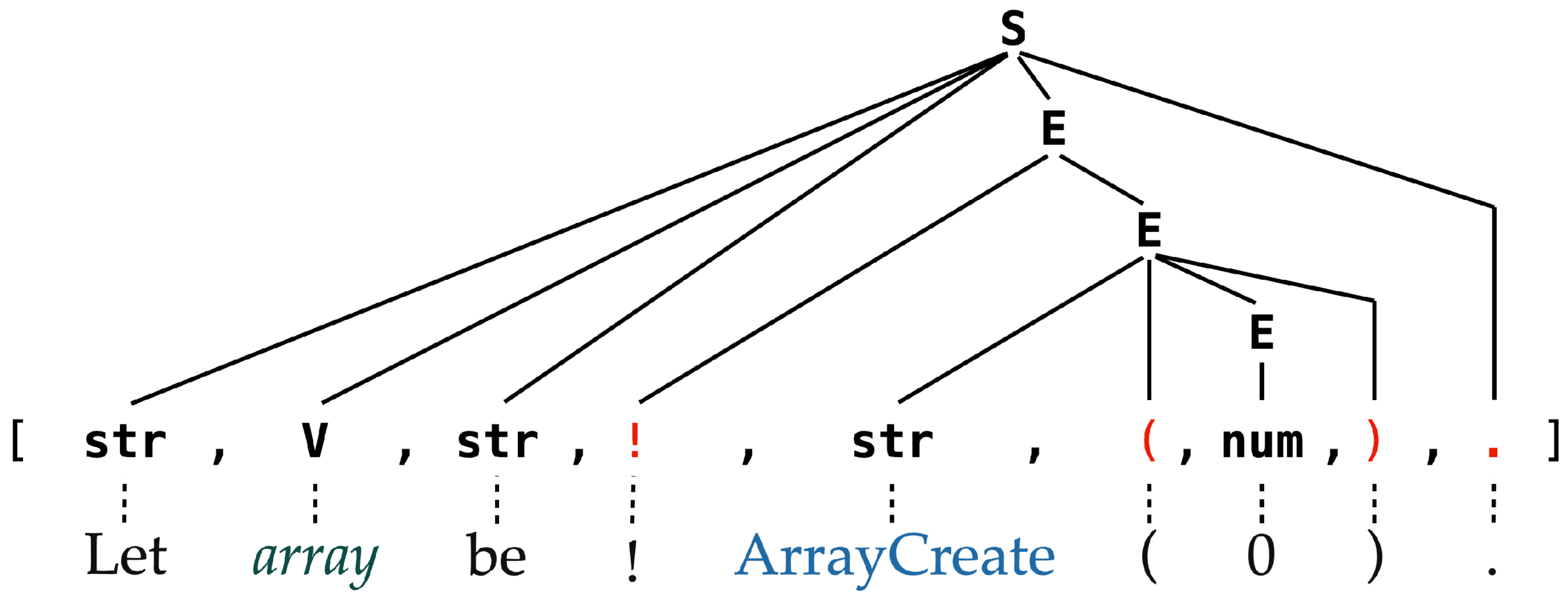


Parsing rules   Conversion Rules

```
S = // statements
  Let ~ V ~ be ~ E ~ .  ^^ ILet

E = // expressions
  ! E                   ^^ EAbruptCheck |
  str ~ ( ~ E ~ )       ^^ ECall        |
  num                   ^^ _.toDouble
```

Simplified compile rules



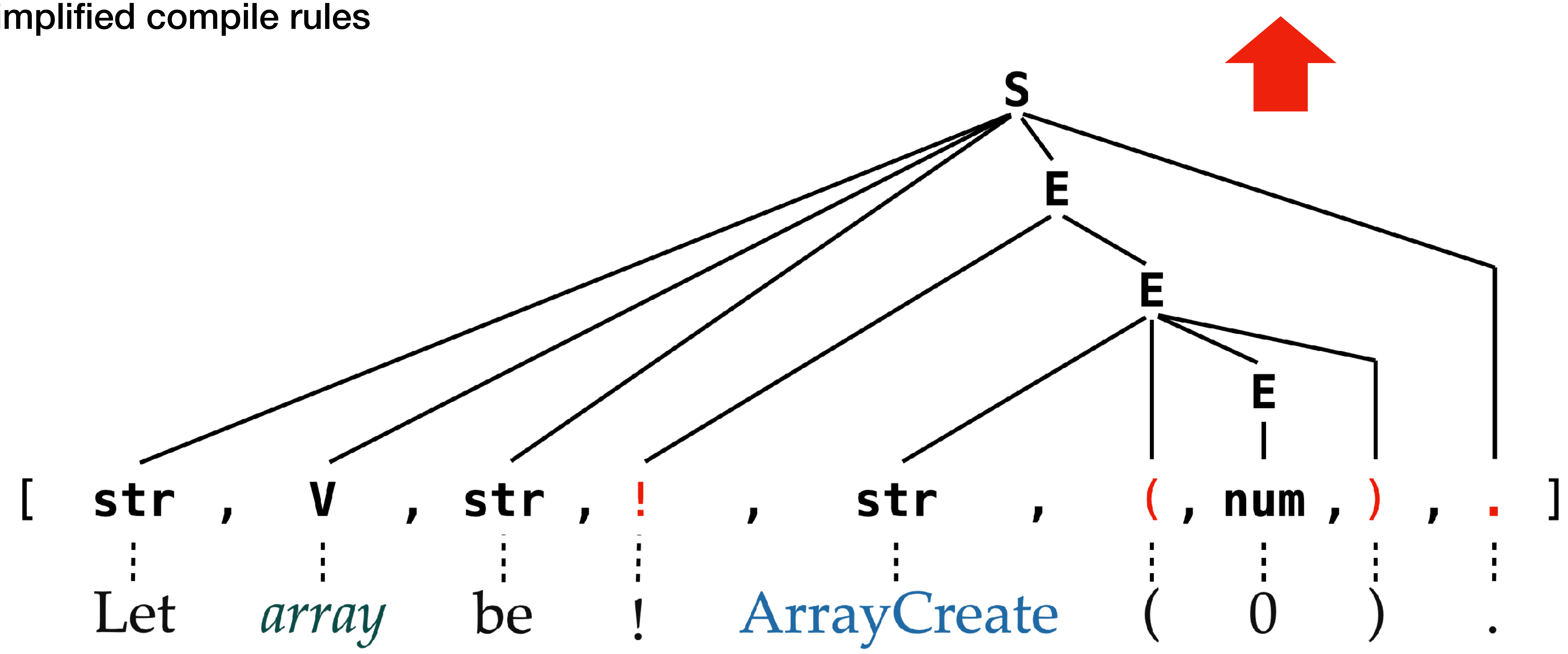
Parsing rules　　　Conversion Rules

```
S = // statements
  Let ~ V ~ be ~ E ~ .  ^^ ILet

E = // expressions
  ! E                   ^^ EAbruptCheck |
  str ~ ( ~ E ~ )       ^^ ECall        |
  num                   ^^ _.toDouble
```

Simplified compile rules

```
ILet(array, EAbruptCheck(
  ECall("ArrayCreate", 0)))
```



Parsing rules Conversion Rules

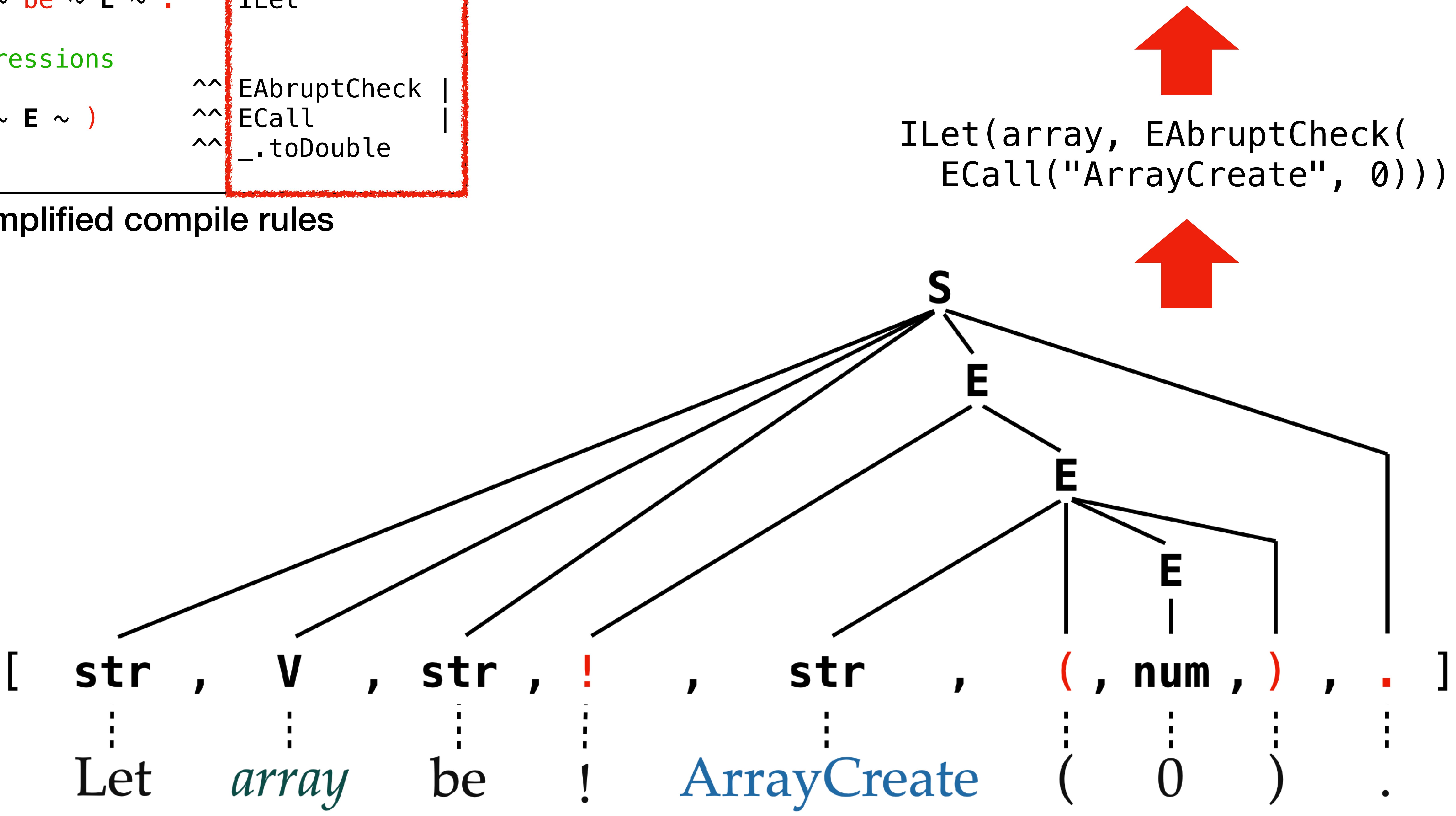
```
S = // statements
  Let ~ V ~ be ~ E ~ .  ^^ ILet

E = // expressions
  ! E                   ^^ EAbruptCheck |
  str ~ ( ~ E ~ )       ^^ ECall        |
  num                   ^^ _.toDouble
```

Simplified compile rules

```
let array = ! (ArrayCreate 0)
```

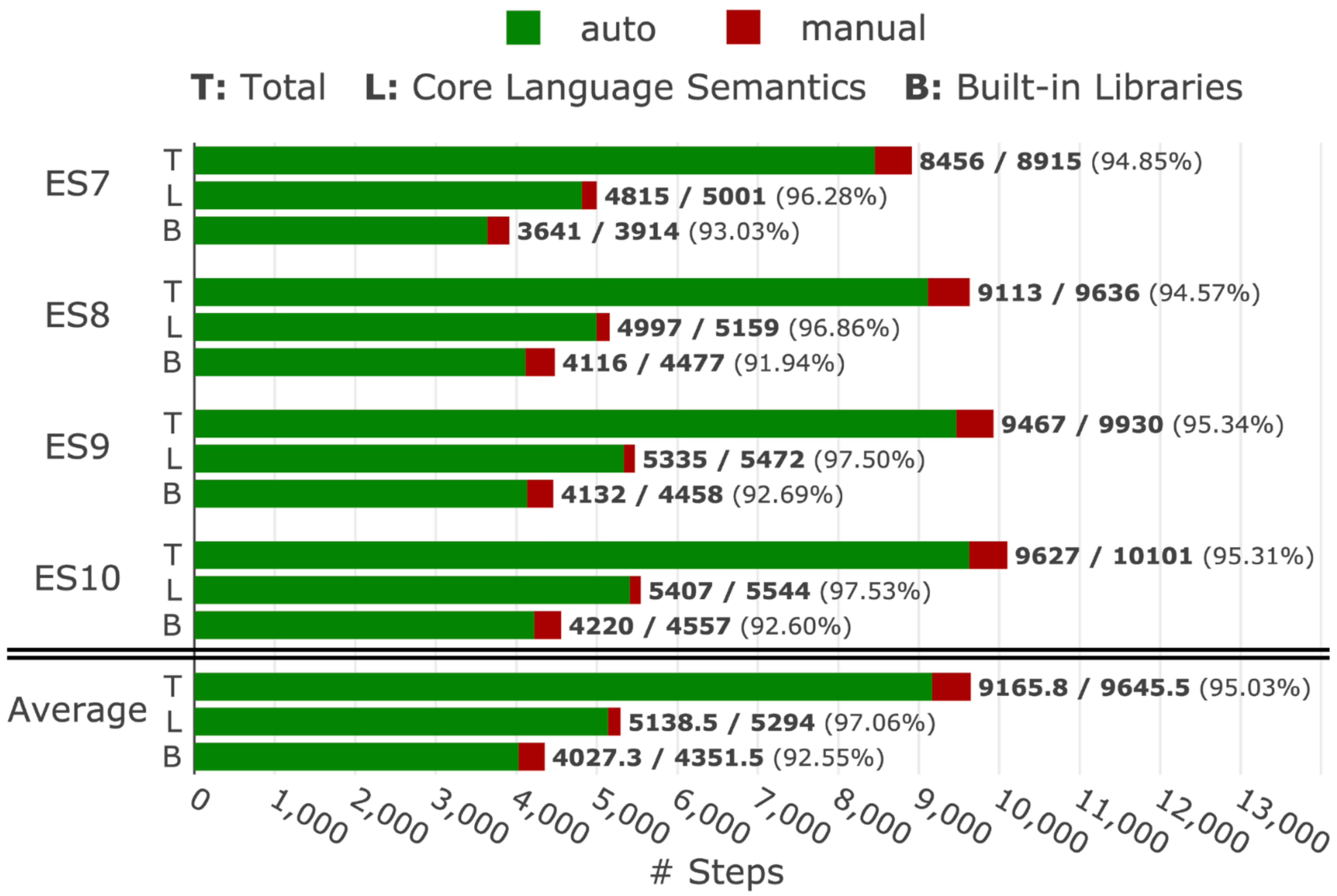
```
ILet(array, EAbruptCheck(
  ECall("ArrayCreate", 0)))
```


S
E
E
E
[str , V , str , ! , str , (, num ,) , .]
Let array be ! ArrayCreate (0) .

Semantics - Evaluation

The number of compile rules

Name	# Rules
Statment	21
Expression	27
Condition	16
Value	11
Type	34
Reference	9
Total	118

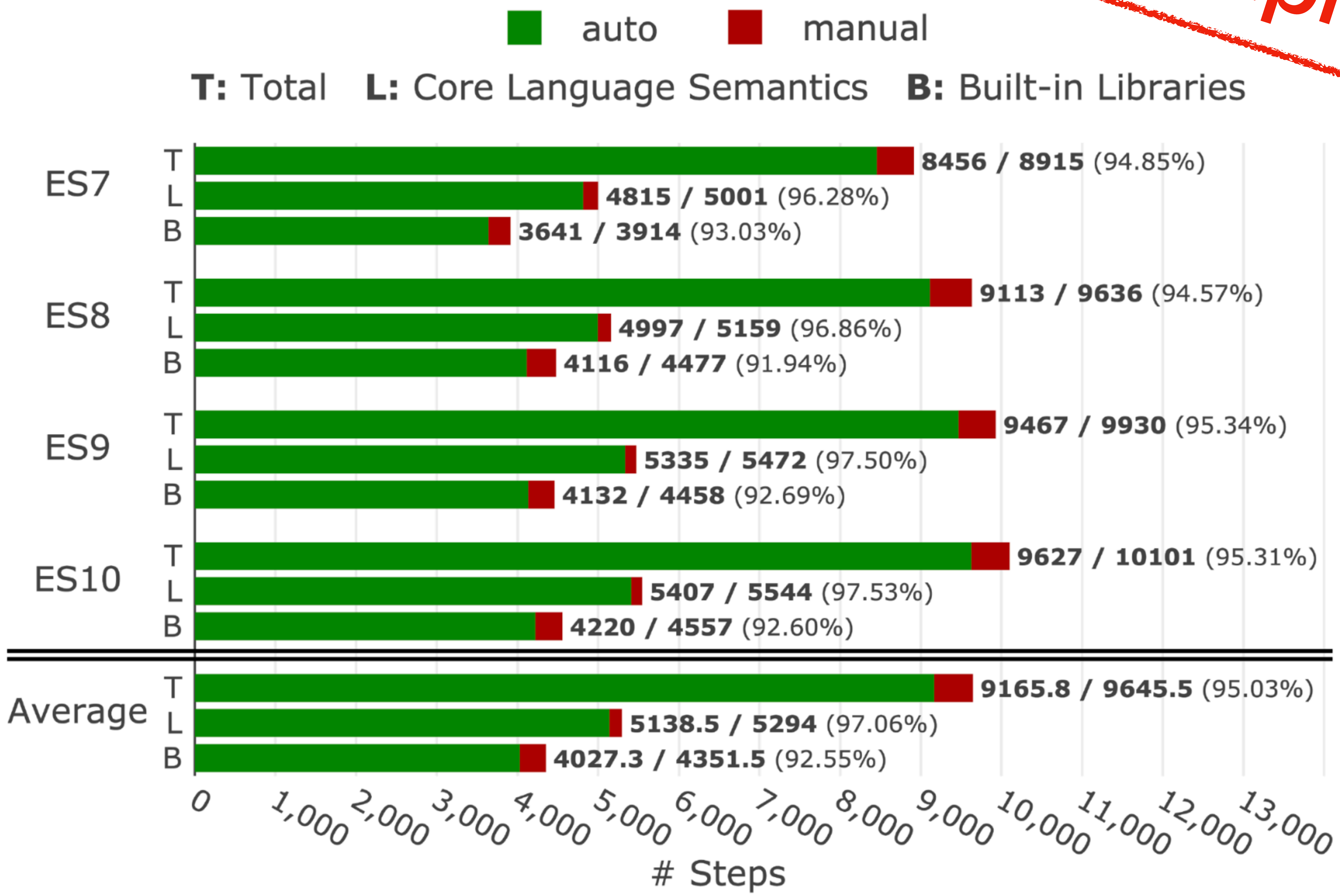


Semantics - Evaluation

≈ 95% Compiled

The number of compile rules

Name	# Rules
Statment	21
Expression	27
Condition	16
Value	11
Type	34
Reference	9
Total	118

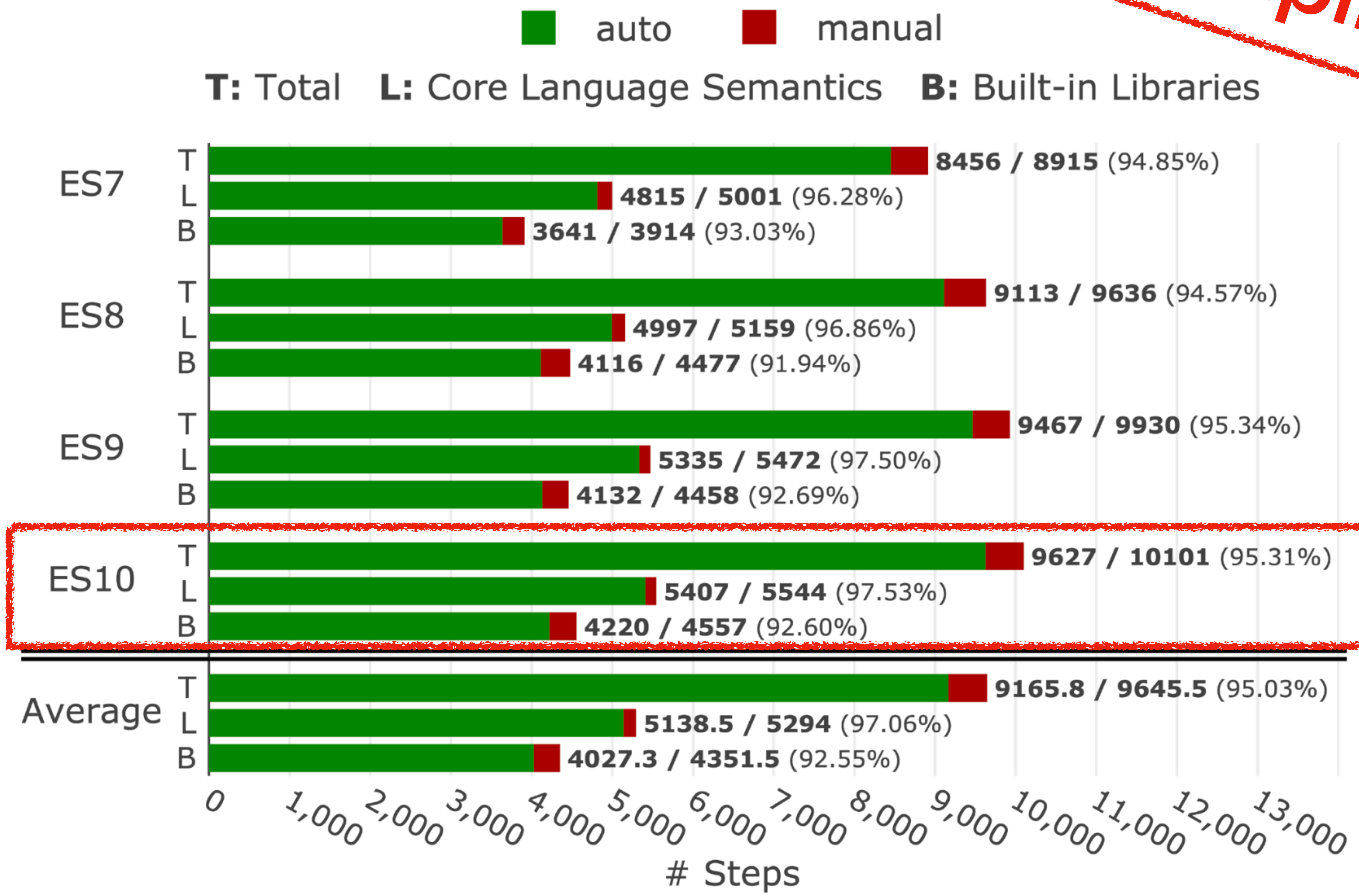


Semantics - Evaluation

≈ 95% Compiled

The number of compile rules

Name	# Rules
Statment	21
Expression	27
Condition	16
Value	11
Type	34
Reference	9
Total	118



Semantics - Evaluation

- **Test262** - Official ECMAScript test suite

16,355 / 18,064 (90.54%)

Name	Feature	Description	Known	Created	Resolved	Existed	# Fails
ES10-1	Iteration	Missing the `async-iterate` case in the assertion of **ForIn/OfHeadEvaluation**	X	2018-02-16	2020-03-25	768 days	1,116
ES10-2	Condition	Ambiguous grammar production for the dangling `else` problem in *IfStatement*	X	2015-06-01	TBD	TBD	1
ES10-3	String	Wrong use of the = operator in **StringGetOwnProperty**	X	2015-06-01	2020-05-07	1,802 days	7
ES10-4	Completion	Unhandling abrupt completion in **Abstract Equality Comparison**	X	2015-06-01	2020-04-28	1,793 days	9
ES10-5	Completion	Unhandling abrupt completion in **Evaluation** of *EqualityExpression*	O	2015-06-01	2019-05-02	1,431 days	2
ES10-6	Await	Passing a value of wrong type to the second parameter of **PromiseResolve**	O	2019-02-27	2019-04-13	45 days	1,294
ES10-7	Function	No semantics of **IsFunctionDefinition** for `function(...){...}`	O	2015-10-30	2020-01-18	1,541 days	306
ES10-8	Function	No semantics of **ExpectedArgumentCount** for the base case of *FormalParameters*	O	2016-11-02	2020-02-20	1,205 days	81
ES10-9	Iteration	Two semantics of **VarScopedDeclarations** for `for await(var x of e){...}`	O	2018-02-16	2019-10-11	602 days	0

292 / 303 (96.37%)

Name	Feature	Description	Known	Created	Resolved	Existed	# Fails
BigInt-1	Expression	Using the wrong variable `oldvalue` instead of `oldValue` in **Evaluation** of *UpdateExpression*	X	2019-09-27	2020-04-23	209 days	533
BigInt-2	Number	Using **ToInt32** instead of **ToUint32** in **Number::unsignedRightShift**	X	2019-09-27	2020-04-23	209 days	2
BigInt-3	Number	Unhandling BigInt values in the **Number** constructor	O	2019-09-27	2019-11-19	53 days	1

Semantics - Evaluation

- **Test262** - Official ECMAScript test suite

16,355 / 18,064
(90.54%)

9 bugs in ES10

18,064 / 18,064
(100.00%)

Name	Feature	Description	Known	Created	Resolved	Existed	# Fails
ES10-1	Iteration	Missing the `async-iterate` case in the assertion of **ForIn/OfHeadEvaluation**	X	2018-02-16	2020-03-25	768 days	1,116
ES10-2	Condition	Ambiguous grammar production for the dangling `else` problem in *IfStatement*	X	2015-06-01	TBD	TBD	1
ES10-3	String	Wrong use of the = operator in **StringGetOwnProperty**	X	2015-06-01	2020-05-07	1,802 days	7
ES10-4	Completion	Unhandling abrupt completion in **Abstract Equality Comparison**	X	2015-06-01	2020-04-28	1,793 days	9
ES10-5	Completion	Unhandling abrupt completion in **Evaluation** of *EqualityExpression*	O	2015-06-01	2019-05-02	1,431 days	2
ES10-6	Await	Passing a value of wrong type to the second parameter of **PromiseResolve**	O	2019-02-27	2019-04-13	45 days	1,294
ES10-7	Function	No semantics of **IsFunctionDefinition** for `function(...){...}`	O	2015-10-30	2020-01-18	1,541 days	306
ES10-8	Function	No semantics of **ExpectedArgumentCount** for the base case of *FormalParameters*	O	2016-11-02	2020-02-20	1,205 days	81
ES10-9	Iteration	Two semantics of **VarScopedDeclarations** for `for await(var x of e){...}`	O	2018-02-16	2019-10-11	602 days	0

292 / 303 (96.37%)

Name	Feature	Description	Known	Created	Resolved	Existed	# Fails
BigInt-1	Expression	Using the wrong variable `oldvalue` instead of `oldValue` in **Evaluation** of *UpdateExpression*	X	2019-09-27	2020-04-23	209 days	533
BigInt-2	Number	Using **ToInt32** instead of **ToUint32** in **Number::unsignedRightShift**	X	2019-09-27	2020-04-23	209 days	2
BigInt-3	Number	Unhandling BigInt values in the **Number** constructor	O	2019-09-27	2019-11-19	53 days	1

Semantics - Evaluation

- **Test262** - Official ECMAScript test suite

16,355 / 18,064 (90.54%)

9 bugs in ES10

18,064 / 18,064 (100.00%)

Name	Feature	Description	Known	Created	Resolved	Existed	# Fails
ES10-1	Iteration	Missing the `async-iterate` case in the assertion of **ForIn/OfHeadEvaluation**	X	2018-02-16	2020-03-25	768 days	1,116
ES10-2	Condition	Ambiguous grammar production for the dangling `else` problem in *IfStatement*	X	2015-06-01	TBD	TBD	1
ES10-3	String	Wrong use of the = operator in **StringGetOwnProperty**	X	2015-06-01	2020-05-07	1,802 days	7
ES10-4	Completion	Unhandling abrupt completion in **Abstract Equality Comparison**	X	2015-06-01	2020-04-28	1,793 days	9
ES10-5	Completion	Unhandling abrupt completion in **Evaluation** of *EqualityExpression*	O	2015-06-01	2019-05-02	1,431 days	2
ES10-6	Await	Passing a value of wrong type to the second parameter of **PromiseResolve**	O	2019-02-27	2019-04-13	45 days	1,294
ES10-7	Function	No semantics of **IsFunctionDefinition** for `function(...){...}`	O	2015-10-30	2020-01-18	1,541 days	306
ES10-8	Function	No semantics of **ExpectedArgumentCount** for the base case of *FormalParameters*	O	2016-11-02	2020-02-20	1,205 days	81
ES10-9	Iteration	Two semantics of **VarScopedDeclarations** for `for await(var x of e){...}`	O	2018-02-16	2019-10-11	602 days	0

292 / 303 (96.37%)

3 bugs in ES.Next

303 / 303 (100.00%)

Name	Feature	Description	Known	Created	Resolved	Existed	# Fails
BigInt-1	Expression	Using the wrong variable `oldvalue` instead of `oldValue` in **Evaluation** of *UpdateExpression*	X	2019-09-27	2020-04-23	209 days	533
BigInt-2	Number	Using **ToInt32** instead of **ToUint32** in **Number::unsignedRightShift**	X	2019-09-27	2020-04-23	209 days	2
BigInt-3	Number	Unhandling BigInt values in the **Number** constructor	O	2019-09-27	2019-11-19	53 days	1

Semantics - Evaluation

All Tests Passed

- **Test262** - Official ECMAScript test suite

16,355 / 18,064 (90.54%)

9 bugs in ES10

18,064 / 18,064 (100.00%)

Name	Feature	Description	Known	Created	Resolved	Existed	# Fails
ES10-1	Iteration	Missing the `async-iterate` case in the assertion of **ForIn/OfHeadEvaluation**	X	2018-02-16	2020-03-25	768 days	1,116
ES10-2	Condition	Ambiguous grammar production for the dangling `else` problem in *IfStatement*	X	2015-06-01	TBD	TBD	1
ES10-3	String	Wrong use of the = operator in **StringGetOwnProperty**	X	2015-06-01	2020-05-07	1,802 days	7
ES10-4	Completion	Unhandling abrupt completion in **Abstract Equality Comparison**	X	2015-06-01	2020-04-28	1,793 days	9
ES10-5	Completion	Unhandling abrupt completion in **Evaluation** of *EqualityExpression*	O	2015-06-01	2019-05-02	1,431 days	2
ES10-6	Await	Passing a value of wrong type to the second parameter of **PromiseResolve**	O	2019-02-27	2019-04-13	45 days	1,294
ES10-7	Function	No semantics of **IsFunctionDefinition** for `function(...){...}`	O	2015-10-30	2020-01-18	1,541 days	306
ES10-8	Function	No semantics of **ExpectedArgumentCount** for the base case of *FormalParameters*	O	2016-11-02	2020-02-20	1,205 days	81
ES10-9	Iteration	Two semantics of **VarScopedDeclarations** for `for await(var x of e){...}`	O	2018-02-16	2019-10-11	602 days	0

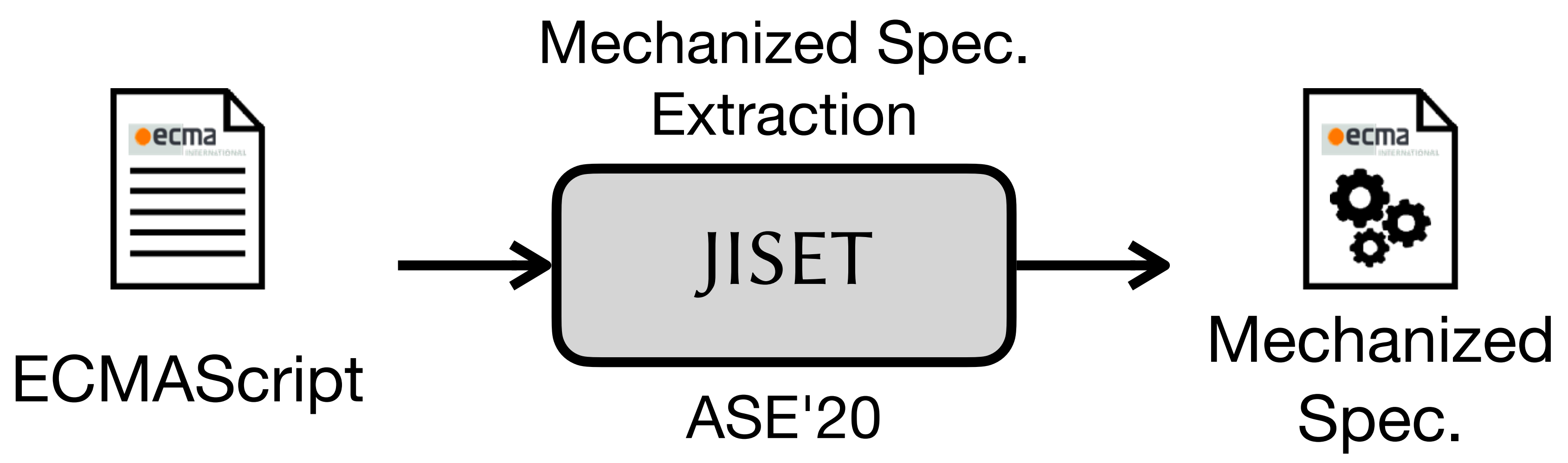
292 / 303 (96.37%)

3 bugs in ES.Next

303 / 303 (100.00%)

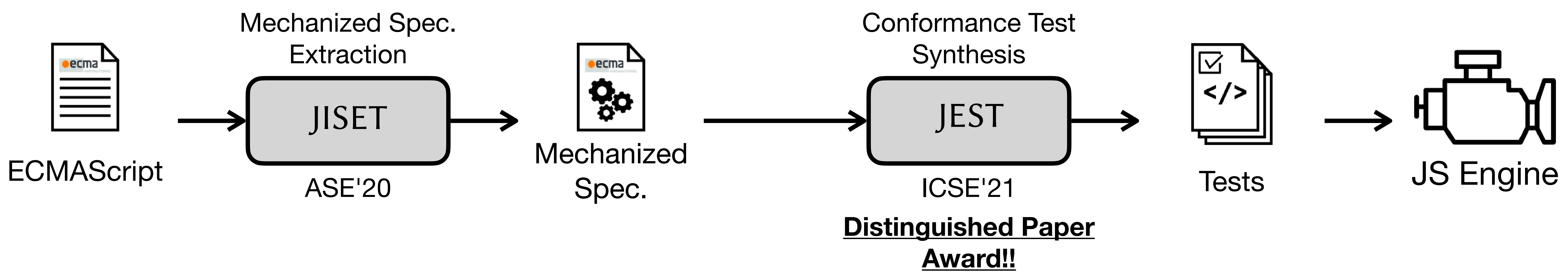
Name	Feature	Description	Known	Created	Resolved	Existed	# Fails
BigInt-1	Expression	Using the wrong variable `oldvalue` instead of `oldValue` in **Evaluation** of *UpdateExpression*	X	2019-09-27	2020-04-23	209 days	533
BigInt-2	Number	Using **ToInt32** instead of **ToUint32** in **Number::unsignedRightShift**	X	2019-09-27	2020-04-23	209 days	2
BigInt-3	Number	Unhandling BigInt values in the **Number** constructor	O	2019-09-27	2019-11-19	53 days	1

Future Work

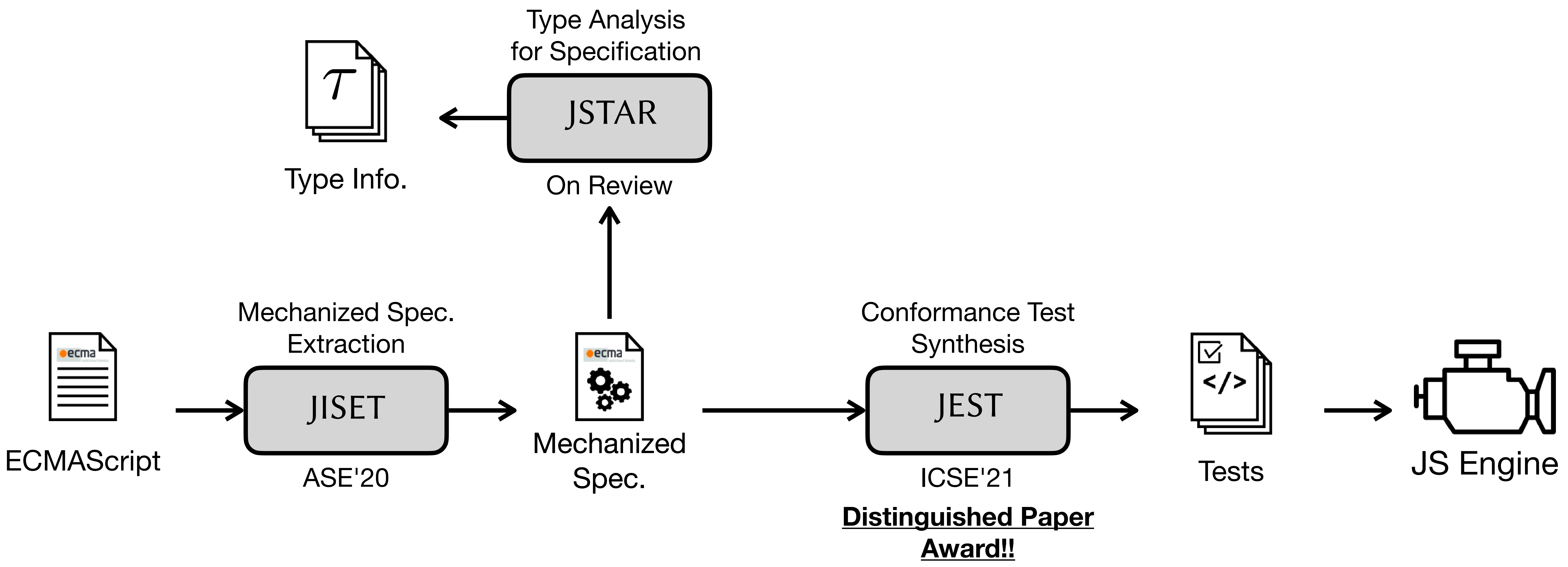


ECMAScript
Mechanized Spec.
Extraction
JISET
ASE'20
Mechanized
Spec.

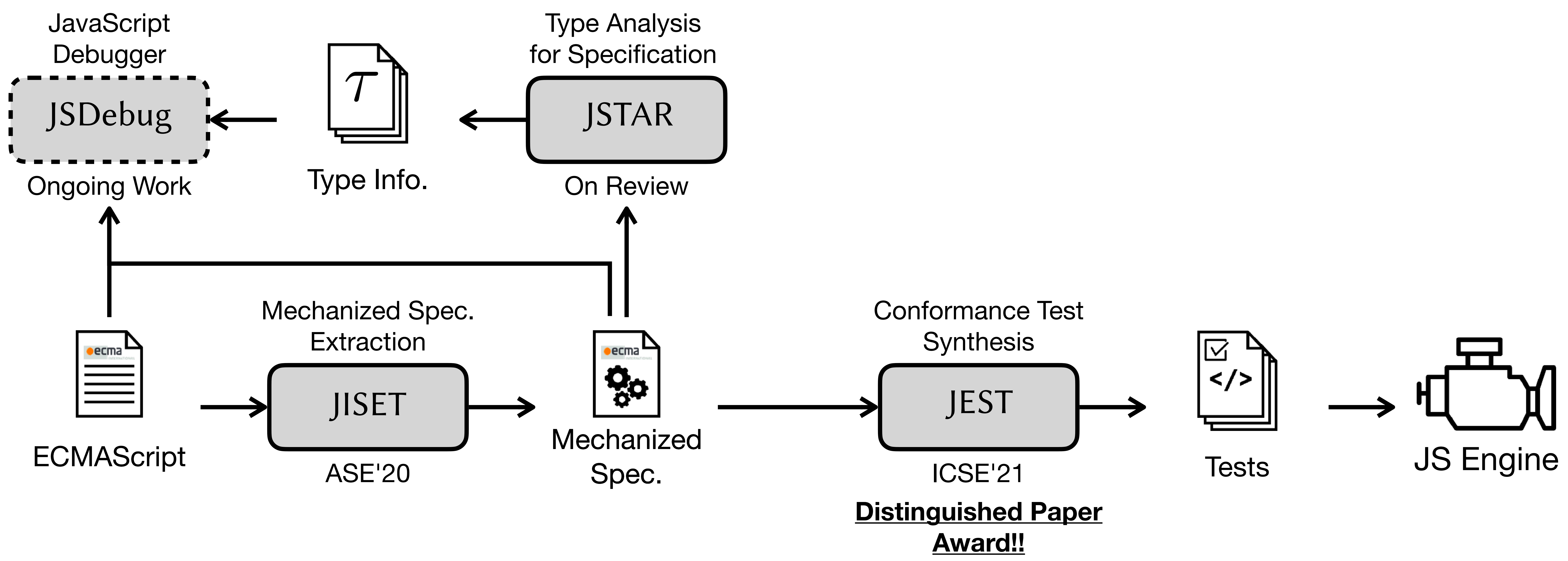
Future Work


ECMAScript
Mechanized Spec.
Extraction
JISET
ASE'20
Mechanized
Spec.
Conformance Test
Synthesis
JEST
ICSE'21
Distinguished Paper
Award!!
Tests
JS Engine

Future Work

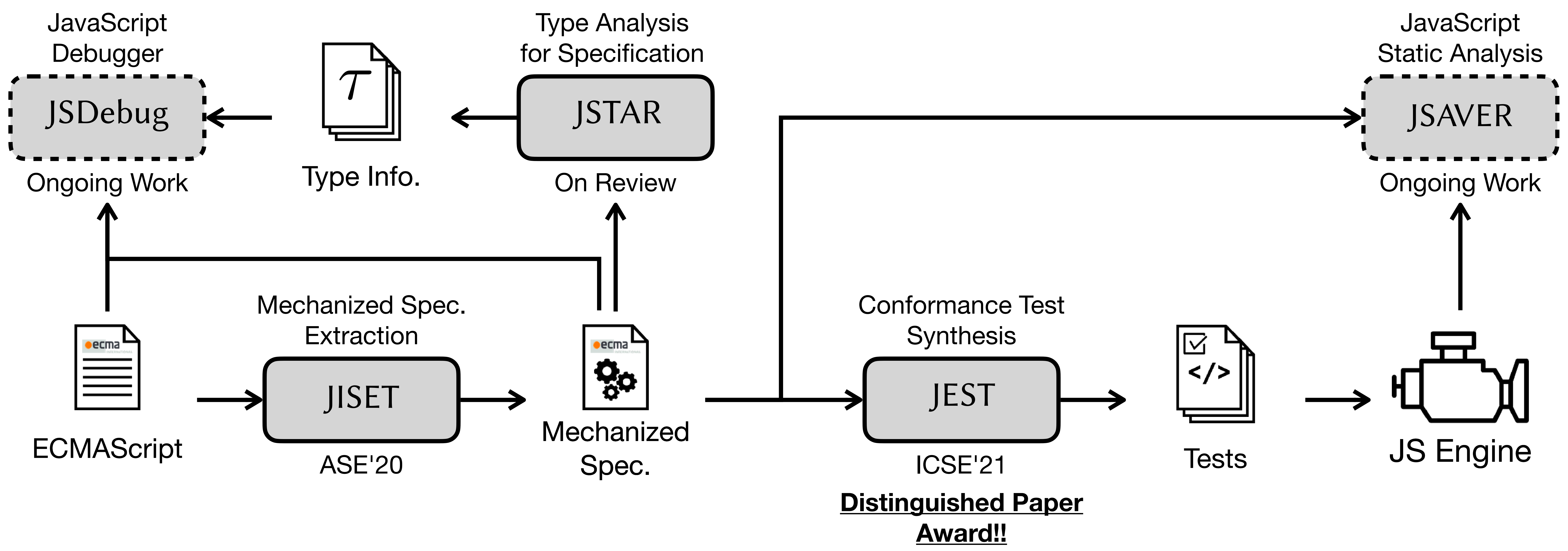

Type Analysis
for Specification
JSTAR
On Review
Type Info.
Mechanized Spec.
Extraction
JISET
ASE'20
ECMAScript
Mechanized
Spec.
Conformance Test
Synthesis
JEST
ICSE'21
Distinguished Paper
Award!!
Tests
JS Engine

Future Work

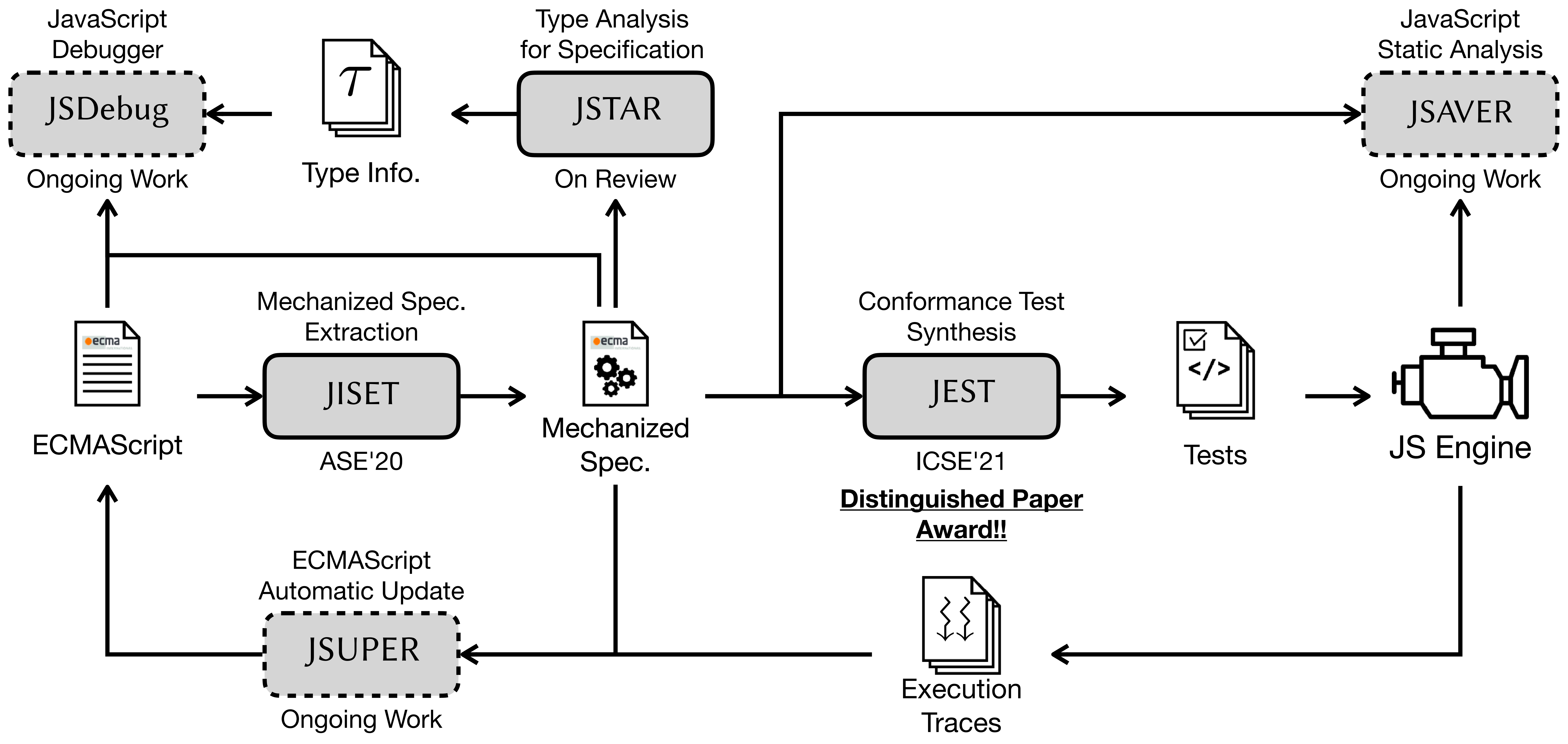


JavaScript Debugger
JSDebug
Ongoing Work
Type Info.
Type Analysis for Specification
JSTAR
On Review
ECMAScript
Mechanized Spec. Extraction
JISET
ASE'20
Mechanized Spec.
Conformance Test Synthesis
JEST
ICSE'21
Distinguished Paper Award!!
Tests
JS Engine

Future Work


JavaScript Debugger
JSDebug
Ongoing Work
Type Info.
Type Analysis for Specification
JSTAR
On Review
JavaScript Static Analysis
JSAVER
Ongoing Work
ECMAScript
Mechanized Spec. Extraction
JISET
ASE'20
Mechanized Spec.
Conformance Test Synthesis
JEST
ICSE'21
Distinguished Paper Award!!
Tests
JS Engine

Future Work



JavaScript Debugger
JSDebug
Ongoing Work
Type Info.
Type Analysis for Specification
JSTAR
On Review
JavaScript Static Analysis
JSAVER
Ongoing Work
ECMAScript
Mechanized Spec. Extraction
JISET
ASE'20
Mechanized Spec.
Conformance Test Synthesis
JEST
ICSE'21
Distinguished Paper Award!!
Tests
JS Engine
ECMAScript Automatic Update
JSUPER
Ongoing Work
Execution Traces