

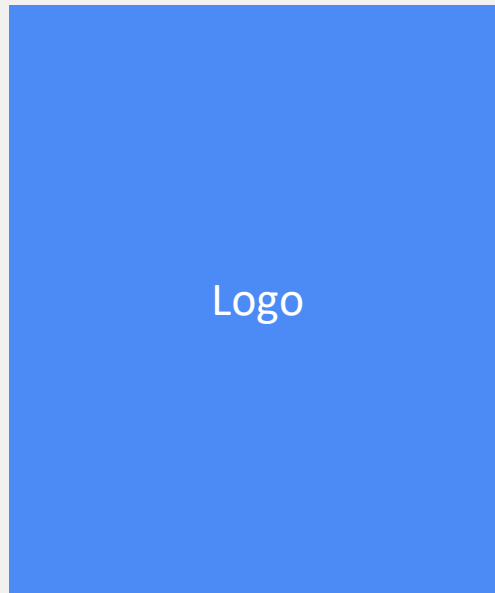
CONTENTS

- General
 - Tools & Plugins
 - Languages
 - Libraries
- Details
 - Implementation
 - Development
 - Architecture
- Live Preview
 - Algorithms and Features

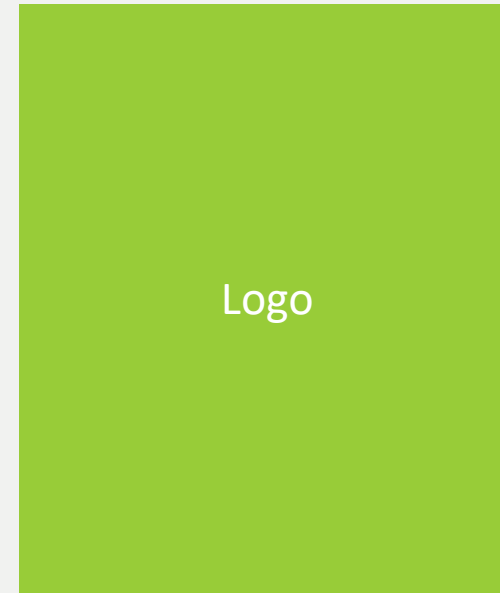
TOOLS & PLUGINS



PhpStorm 2017.2.4



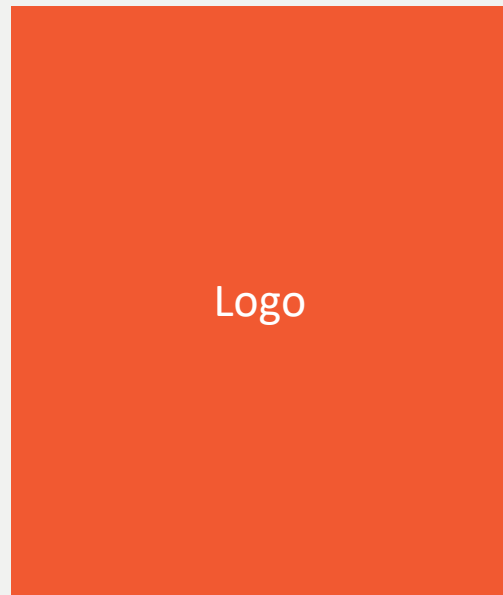
Chrome



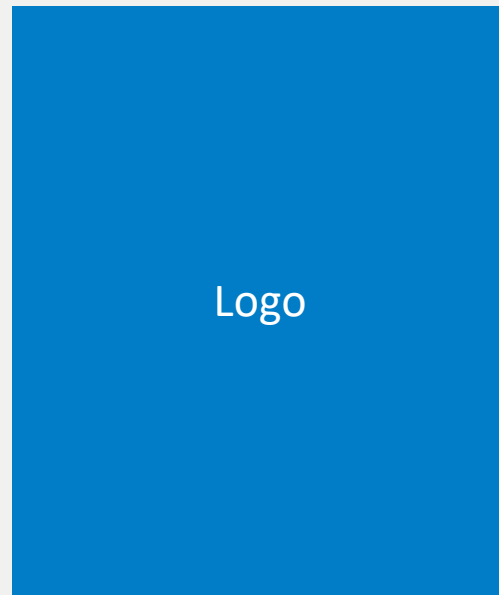
JsTestDriver

Fig. 1.1: Logos of Chrome-Browser, IDE PhpStorm and Unit-Test -Environment JsTestDriver [P2, P3]

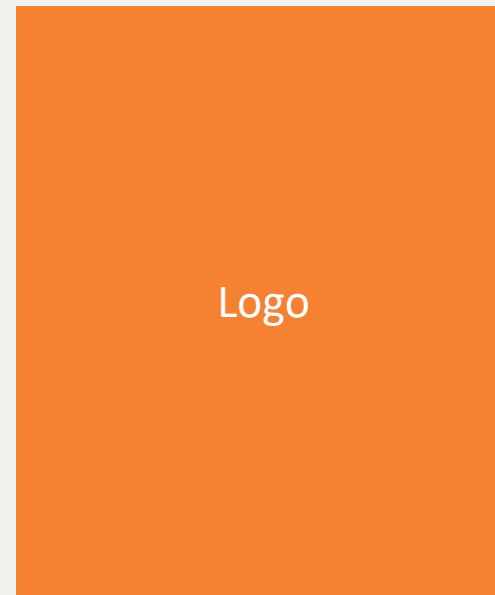
LANGUAGES



HTML 5



CSS 3



JavaScript
(ECMAScript 5.1)

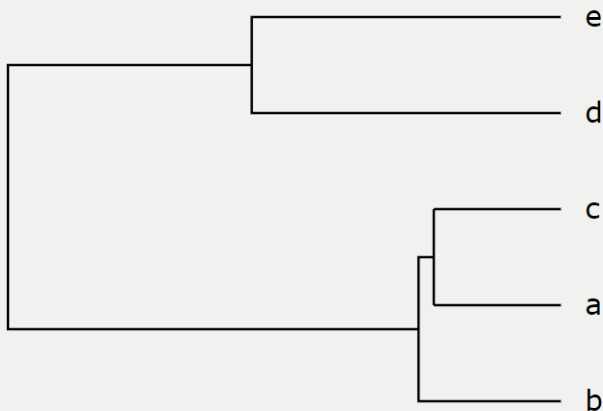
Fig. 1.2: Logos of HTML 5, CSS 3 and JavaScript [P4]

jQuery

namespace.js

Knockout

FileSaver.js



jsPhyloSVG

Fig. 1.4: Phylogenetic Tree
created with jsPhyloSVG

Fig. 1.3: Logos of used libraries [P5-P7]

ALBERT LUDWIGS UNIVERSITY FREIBURG
DEPARTMENT OF COMPUTER SCIENCE
BIONFORMATICS GROUP FREIBURG

DETAILS

IMPLEMENTATION, DEVELOPMENT
& ARCHITECTURE

IMPLEMENTATION

OBJECT-ORIENTED PROGRAMMING

```
1 (function() {  
2   // namespace name, "static" methods  
3   namespace("needlemanWunsch", NeedlemanWunsch);  
xx   ...  
12  function NeedlemanWunsch() {  
13    // inheritance  
14    alignmentInstance = new bases.alignment.Alignment(this);  
15  
16    this.setInput = alignmentInstance.setLinearAlignmentInput;  
17    this.compute = alignmentInstance.compute;  
18    this.getOutput = alignmentInstance.getOutput;  
19  
20    // public methods (available through an instance)  
21    this.getSuperclass = getSuperclass;  
22  }  
xx   ...  
89 }());
```

Code 2.1: OOP-Simulation

IMPLEMENTATION

OBJECT-ORIENTED PROGRAMMING

```
1 (function() {  
2   // namespace name, "static" methods  
3   namespace("needlemanWunsch", NeedlemanWunsch);  
xx  
12  function NeedlemanWunsch() {  
xx    ...  
20    // public methods (available through an instance)  
21    this.getSuperclass = getSuperclass;  
22  }  
23  
24  function a() { // private, because not defined in constructor  
xx    ...  
25  }  
xx  ...  
86  function getSuperclass() { // public, because in constructor  
87    return alignmentInstance;  
88  }  
89 }());
```

Code 2.1: OOP-Simulation

DEVELOPMENT EXTENSIVELY TESTED

- 52 Unit-Tests
 - step-by-step PDF-files
 - implementations with **JsTestDriver**

T-Coffee

$$\begin{aligned} EL_{1,1}^{a,c} &= L_{1,1}^{a,c} + \sum_{x \in \{b\}} \sum_{k \in \{1,2\}} \min(L_{1,k}^{a,x}, L_{k,1}^{x,b}) \\ &= \frac{200}{3} + \min(L_{1,1}^{a,b}, L_{1,1}^{b,c}) + \min(L_{1,2}^{a,b}, L_{2,1}^{b,c}) \\ &= \frac{200}{3} + \min(100, 0) + \min(0, 50) \end{aligned}$$

$$\begin{aligned} EL_{2,2}^{a,c} &= L_{2,2}^{a,c} + \sum_{x \in \{b\}} \sum_{k \in \{1,2\}} \min(L_{2,k}^{a,x}, L_{k,2}^{x,b}) \\ &= \frac{200}{3} + \min(L_{2,1}^{a,b}, L_{1,2}^{b,c}) + \min(L_{2,2}^{a,b}, L_{2,2}^{b,c}) \\ &= \frac{200}{3} + \min(0, 50) + 0 \end{aligned}$$

PDF

Fig. 2.1:
Excerpt from unit-test
Notredame-Higgins-Heringa

ARCHITECTURE

LOADING

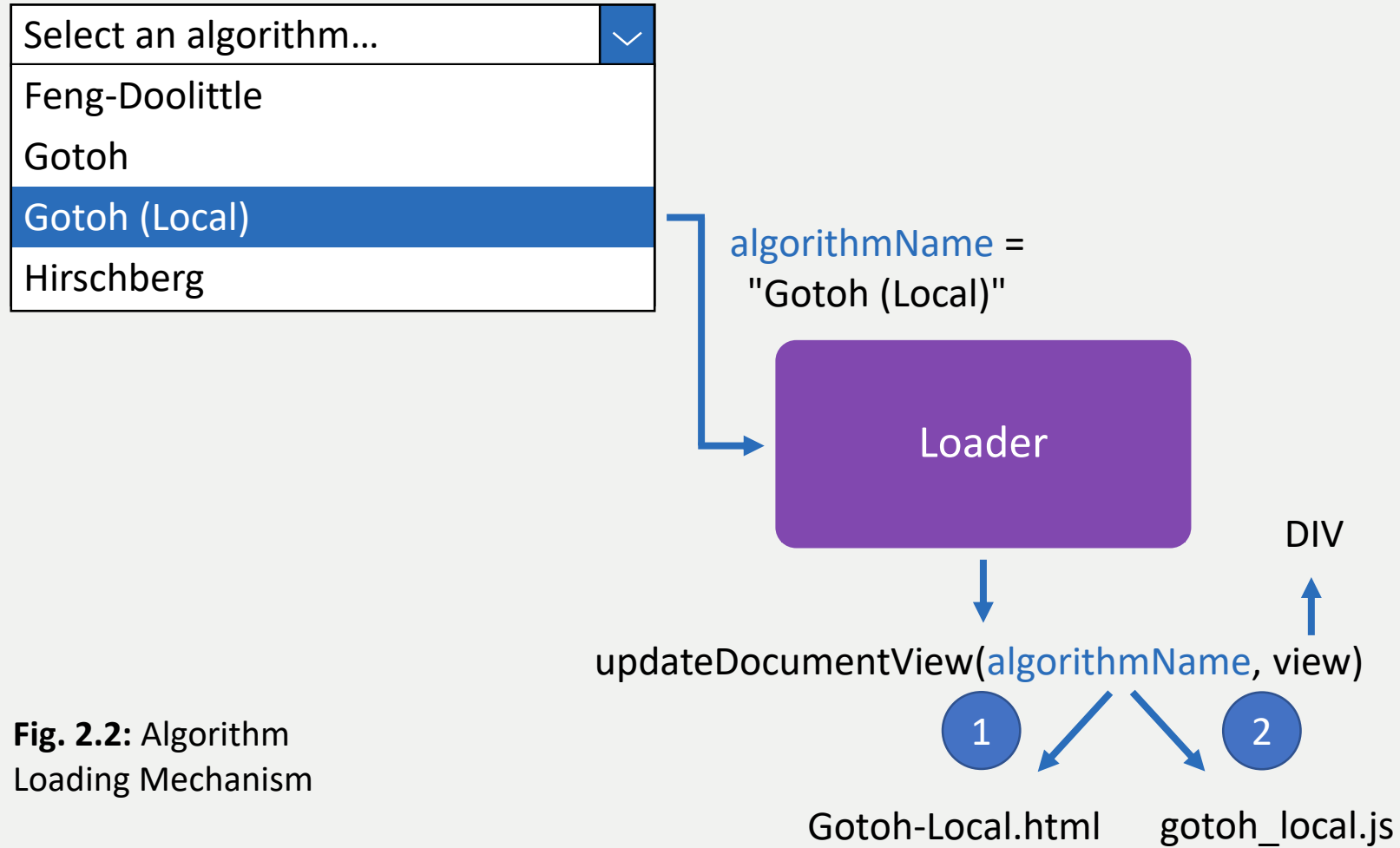


Fig. 2.2: Algorithm Loading Mechanism

ARCHITECTURE

SEPARATION OF ALGORITHM AND INTERFACE LOGIC

<i>D</i>		<i>A</i> ₁	<i>A</i> ₂
	0	-2	-4
<i>A</i> ₁	-2	1	-1
<i>C</i> ₂	-4	-1	0
<i>G</i> ₃	-6	-3	-2
Score: -2			

Fig. 2.4: Highlighted Needleman-Wunsch matrix

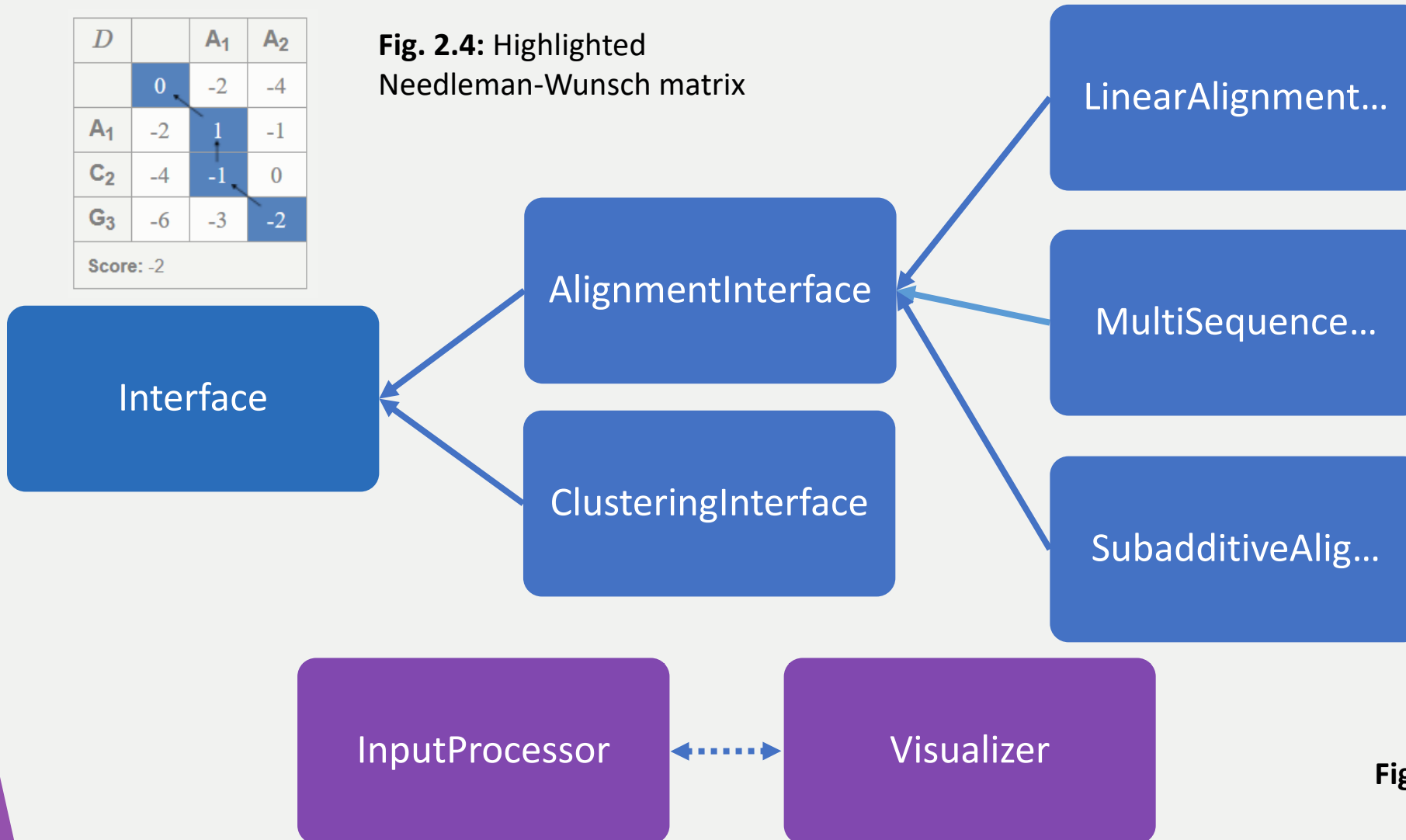


Fig. 2.3: Partial class diagram

ARCHITECTURE

SEPARATION OF ALGORITHM AND INTERFACE LOGIC

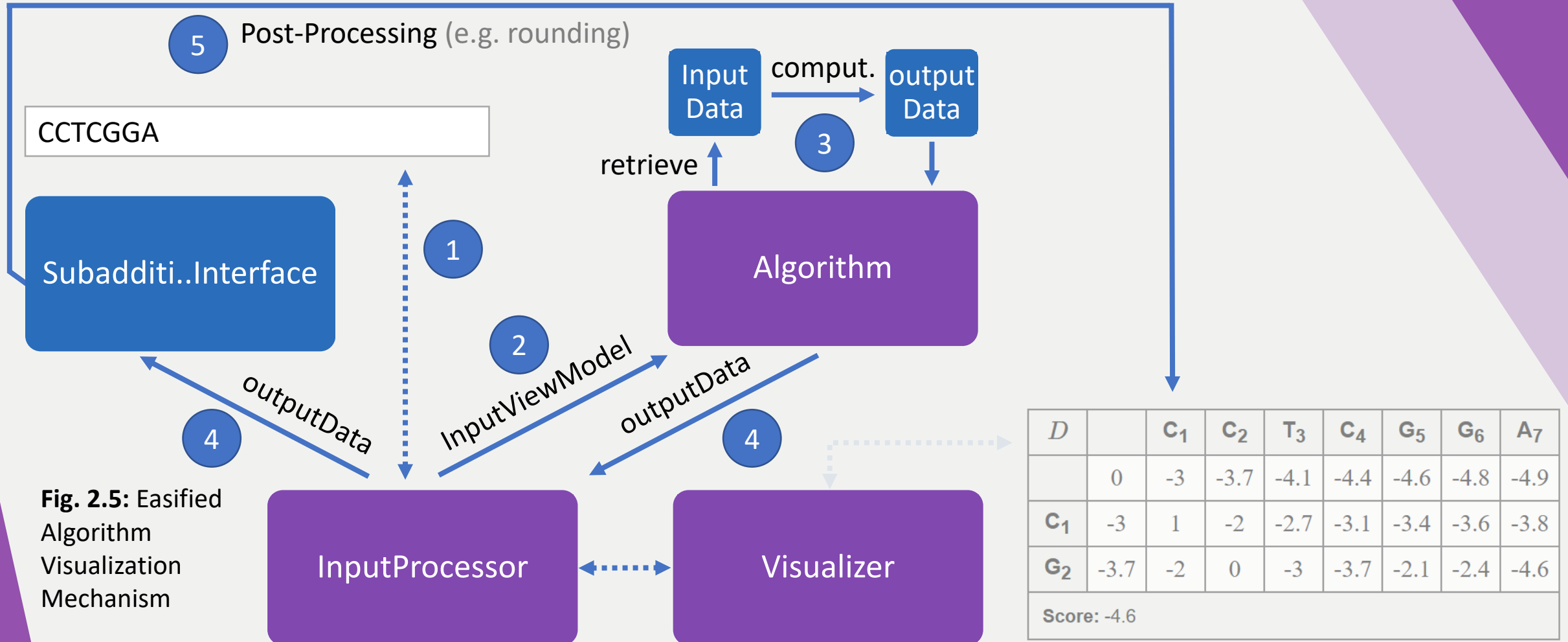


Fig. 2.5: Easified Algorithm Visualization Mechanism

ARCHITECTURE

SEPARATION OF ALGORITHM AND INTERFACE LOGIC

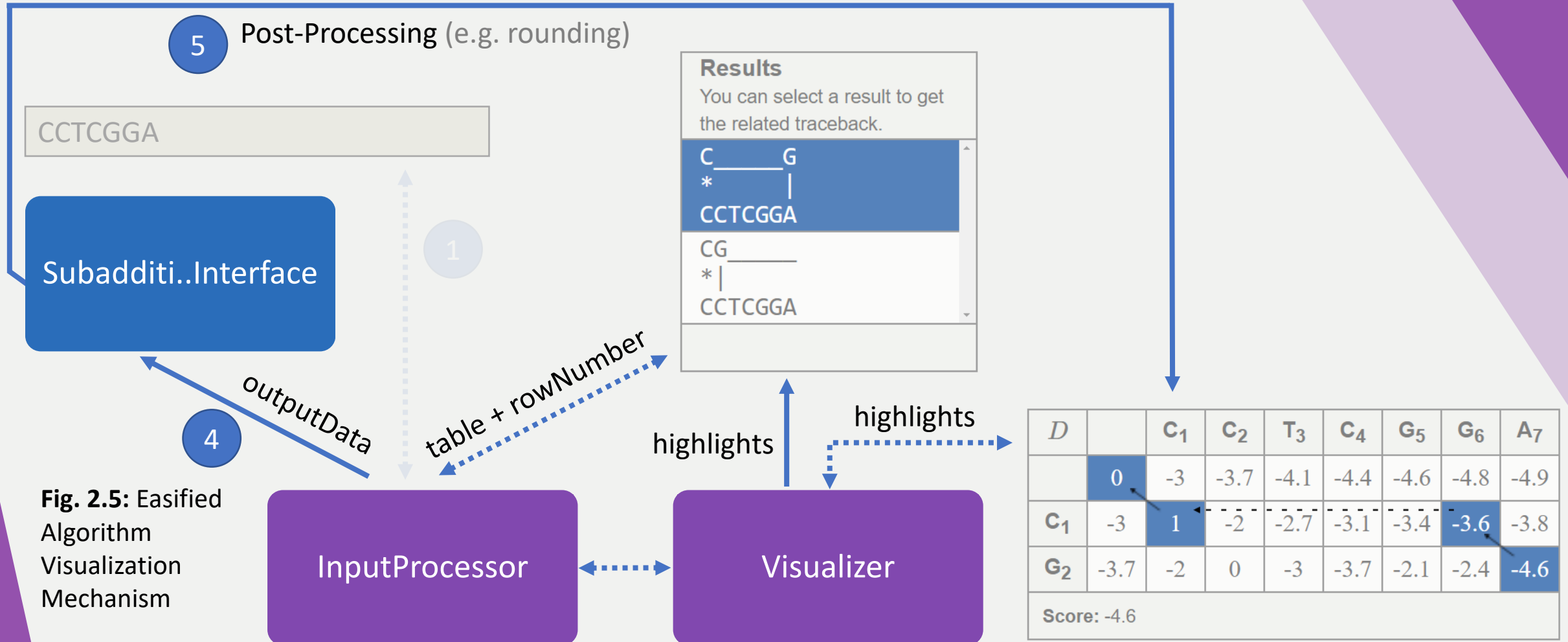


Fig. 2.5: Easified Algorithm Visualization Mechanism

ARCHITECTURE

ADVANTAGES – IDEA BEHIND

jQuery

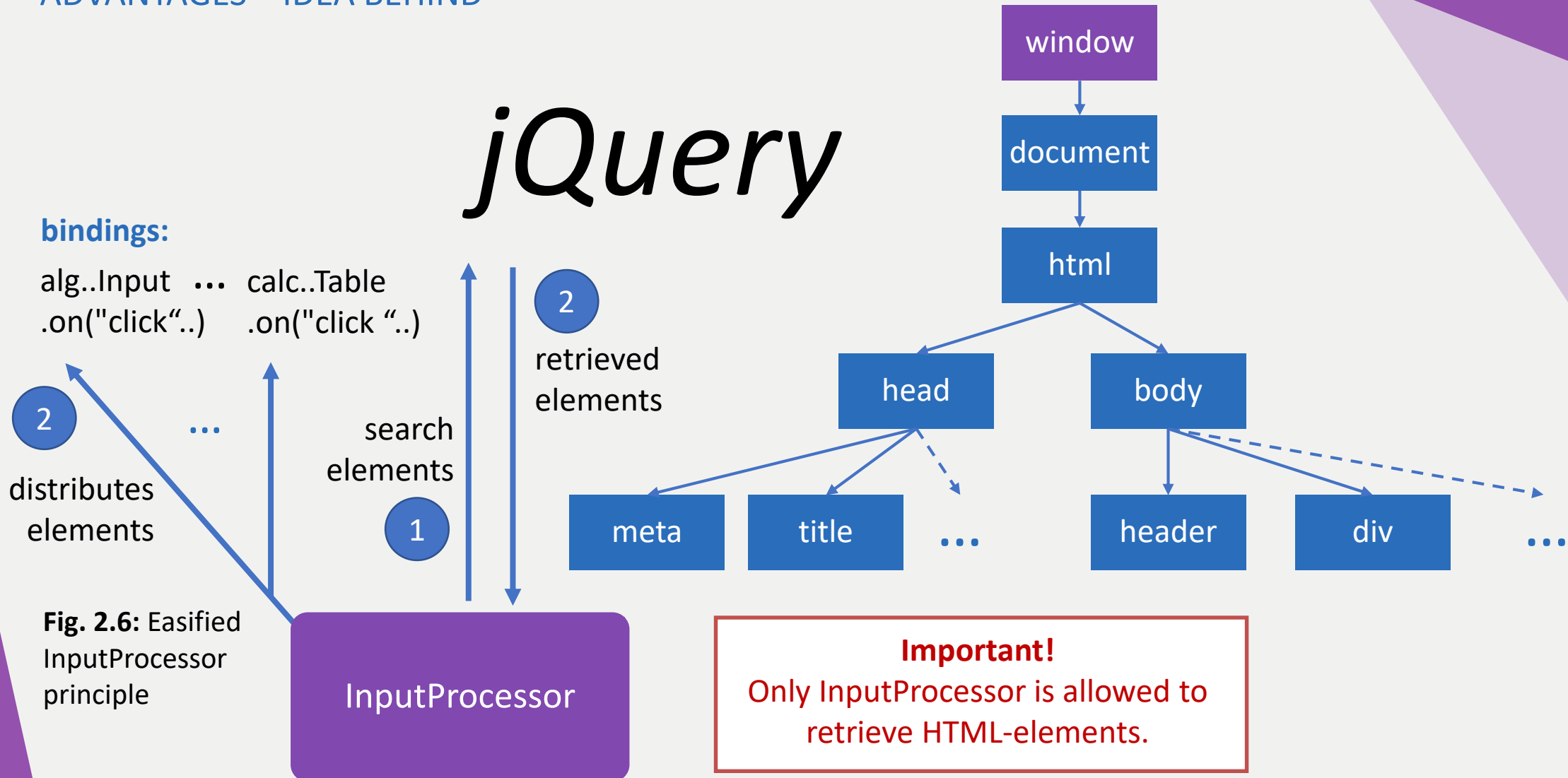


Fig. 2.6: Easified
InputProcessor
principle

SOURCES

- [1] 2017.12.09, jQuery Traversing,
URL: https://www.w3schools.com/jquery/jquery_traversing.asp