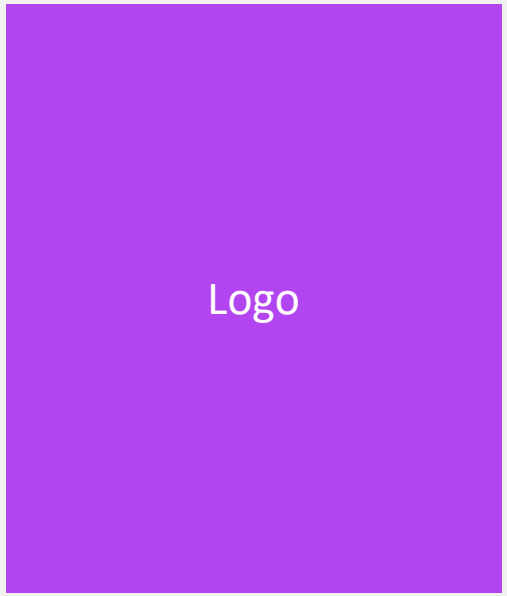
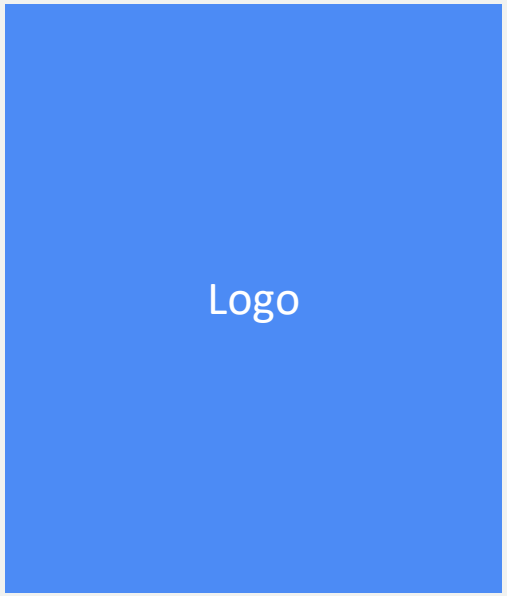
Contents

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

Tools & Plugins



PhpStorm 2017.2.4

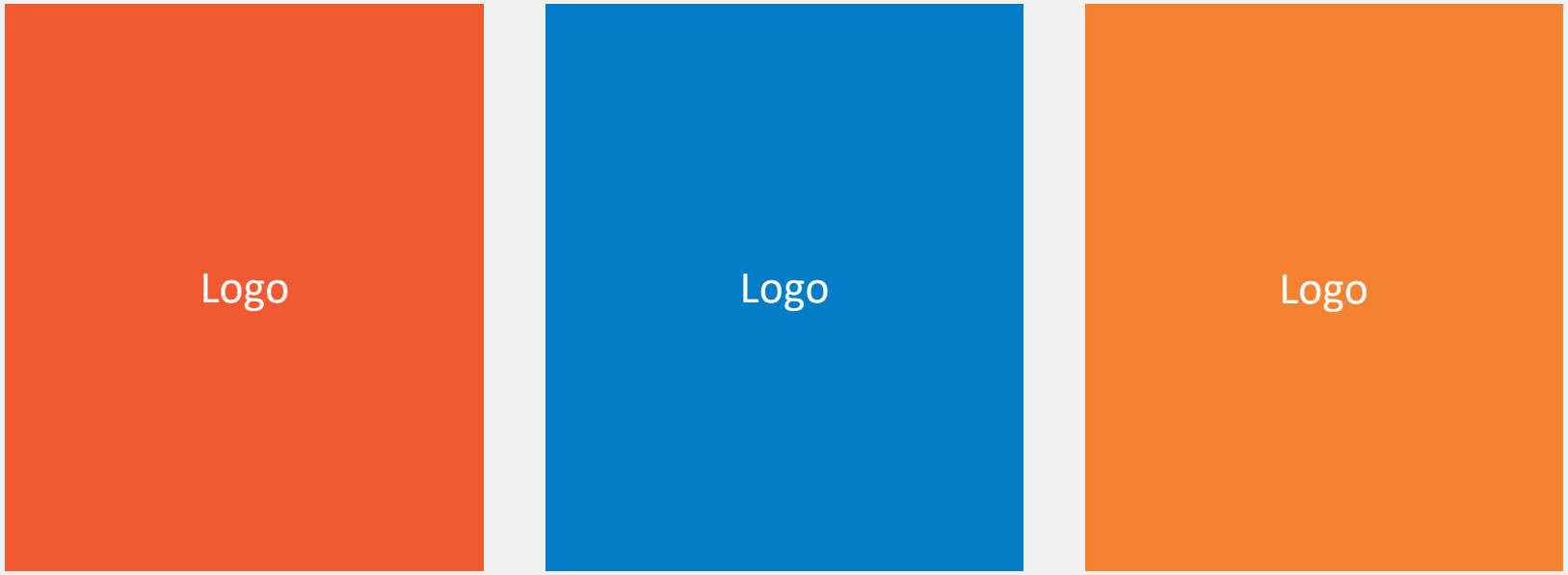


Chrome

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

Knockout

namespace.js

FileSaver.js

*js*PhyloSVG

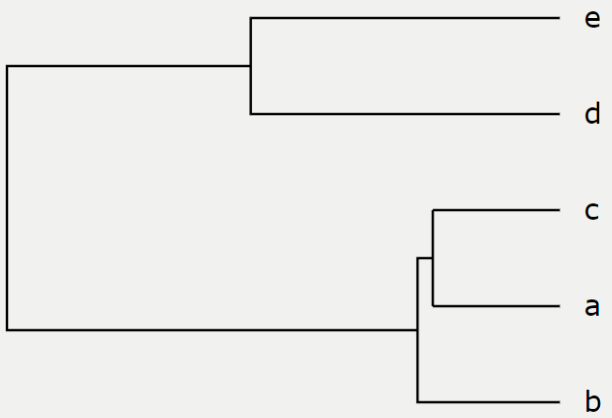


Fig. 1.4: Phylogenetic Tree created with jsPhyloSVG

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, "public 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, "public 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

$$EL_{1,1}^{a,c} = L_{1,1}^{a,c} + \sum_{x\in\{b\}} \sum_{k\in\{1,2\}} \min\left(L_{1,k}^{a,x}, L_{k,1}^{x,b}\right)$$

$$= \frac{200}{3} + \min\left(L_{1,1}^{a,b}, L_{1,1}^{b,c}\right) + \min\left(L_{1,2}^{a,b}, L_{2,1}^{b,c}\right)$$

$$= \frac{200}{3} + \min(100, 0) + \min(0, 50)$$

$$EL_{2,2}^{a,c} = L_{2,2}^{a,c} + \sum_{x\in\{b\}} \sum_{k\in\{1,2\}} \min\left(L_{2,k}^{a,x}, L_{k,2}^{x,b}\right)$$

$$= \frac{200}{3} + \min\left(L_{2,1}^{a,b}, L_{1,2}^{b,c}\right) + \min\left(L_{2,2}^{a,b}, L_{2,2}^{b,c}\right)$$

$$= \frac{200}{3} + \min(0,50) + 0$$

PDF

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

ARCHITECTURE

LOADING

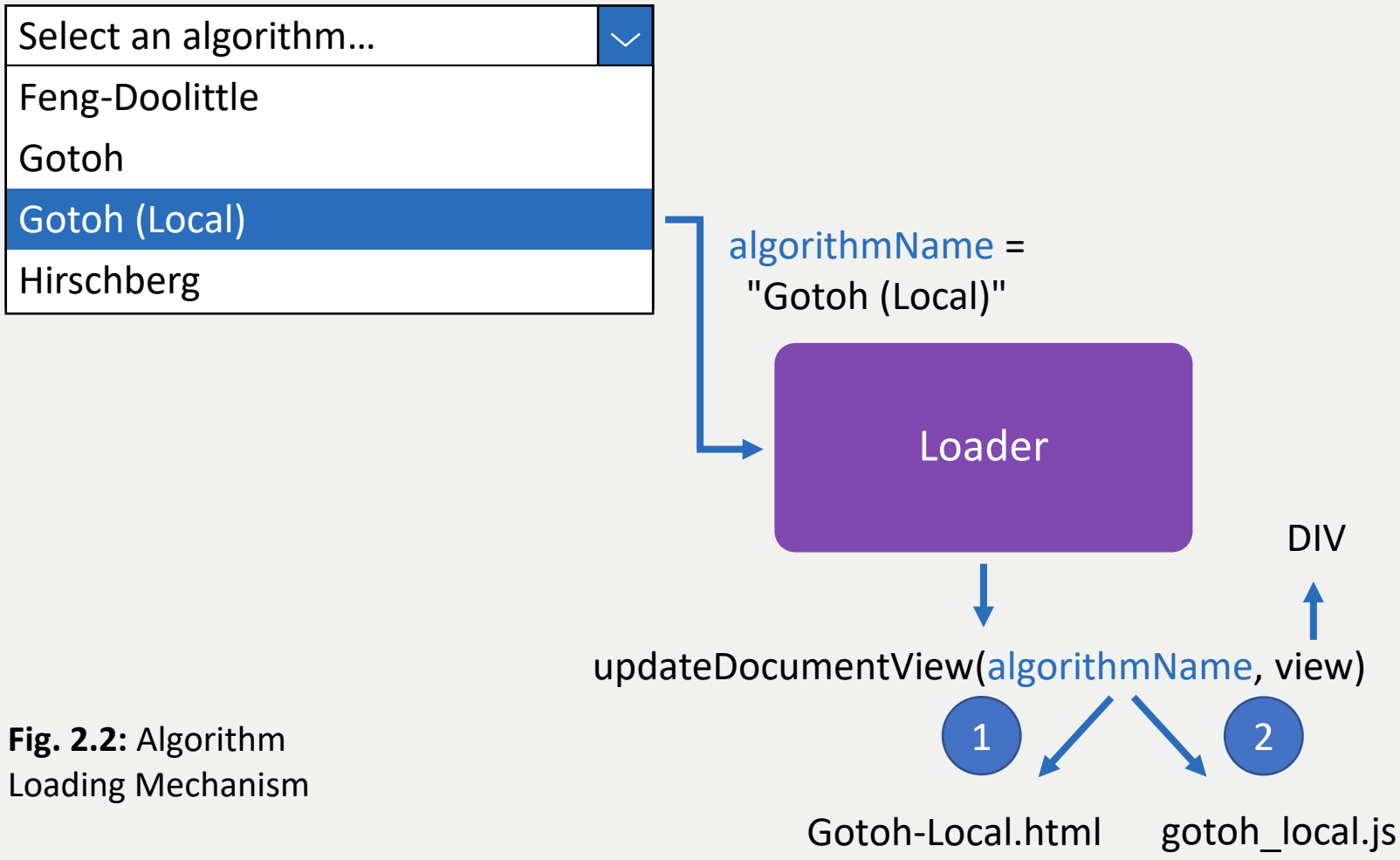


Fig. 2.2: Algorithm Loading Mechanism

ARCHITECTURE

SEPARATION OF ALGORITHM AND INTERFACE LOGIC

D		A_1	A_2
	0	-2	-4
A_1	-2	1	-1
C_2	-4	-1	0
G_3	-6	-3	-2
Score: -2			

Fig. 2.4: Highlighted Needleman-Wunsch matrix

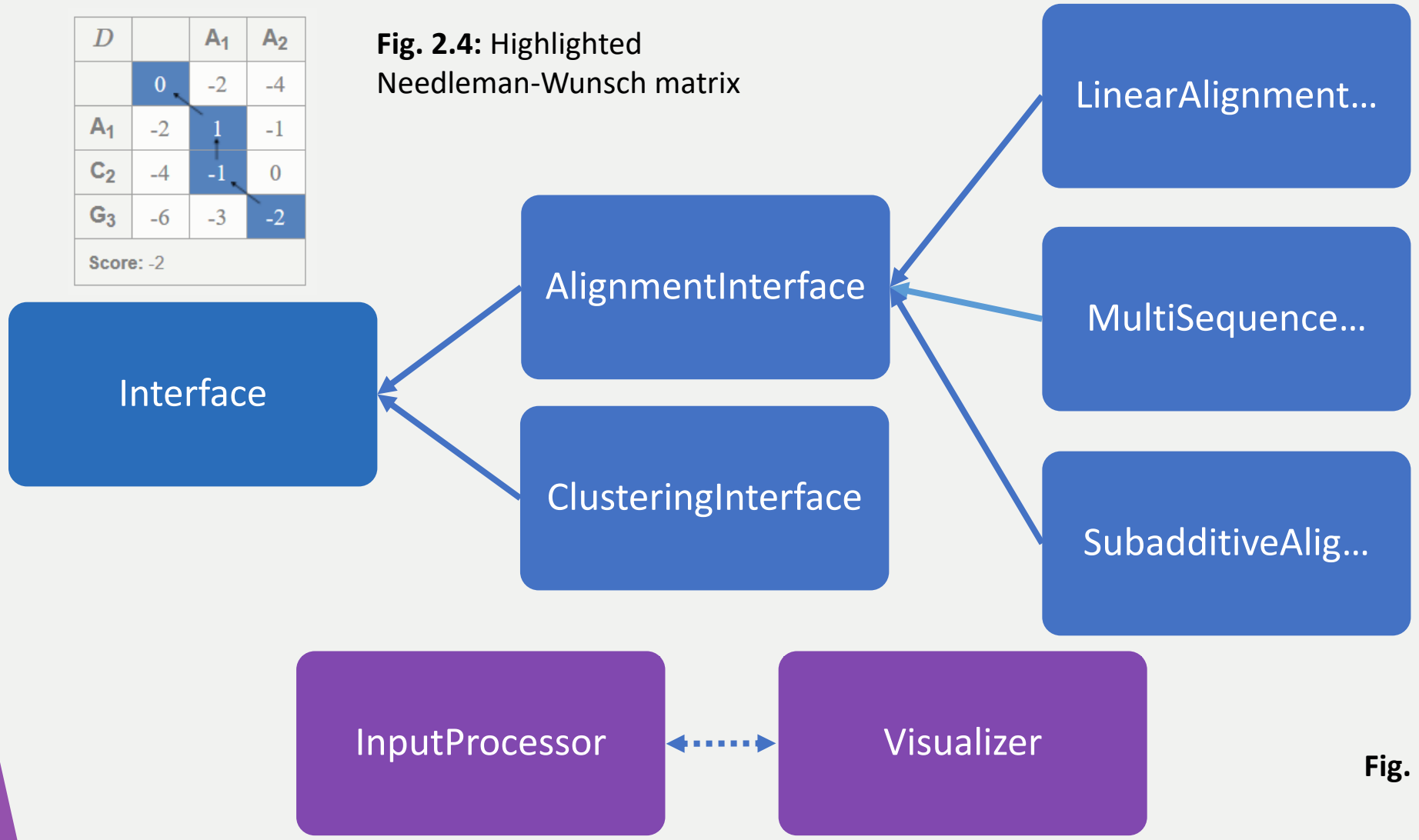


Fig. 2.3: Partial class diagram

ARCHITECTURE

SEPARATION OF ALGORITHM AND INTERFACE LOGIC

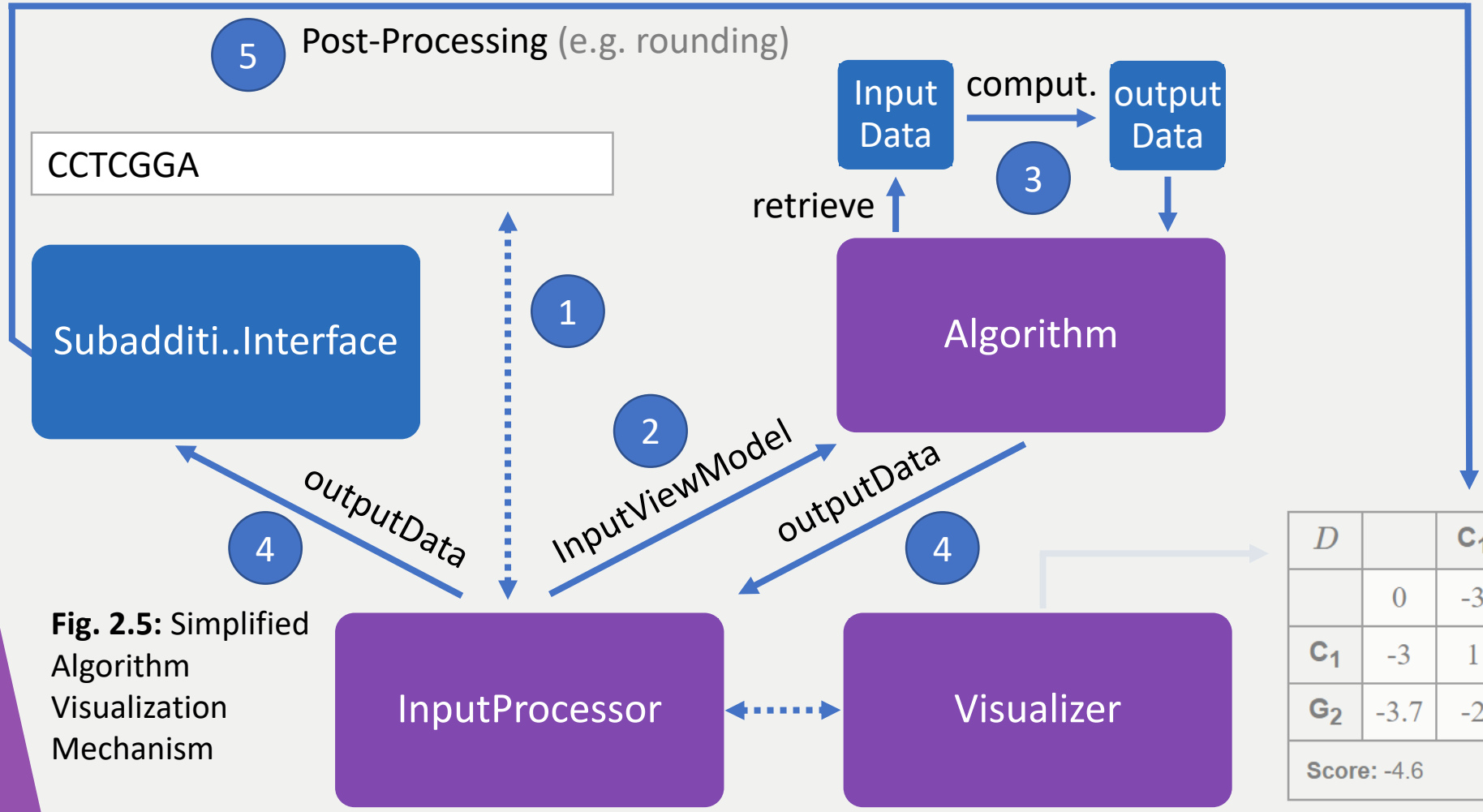
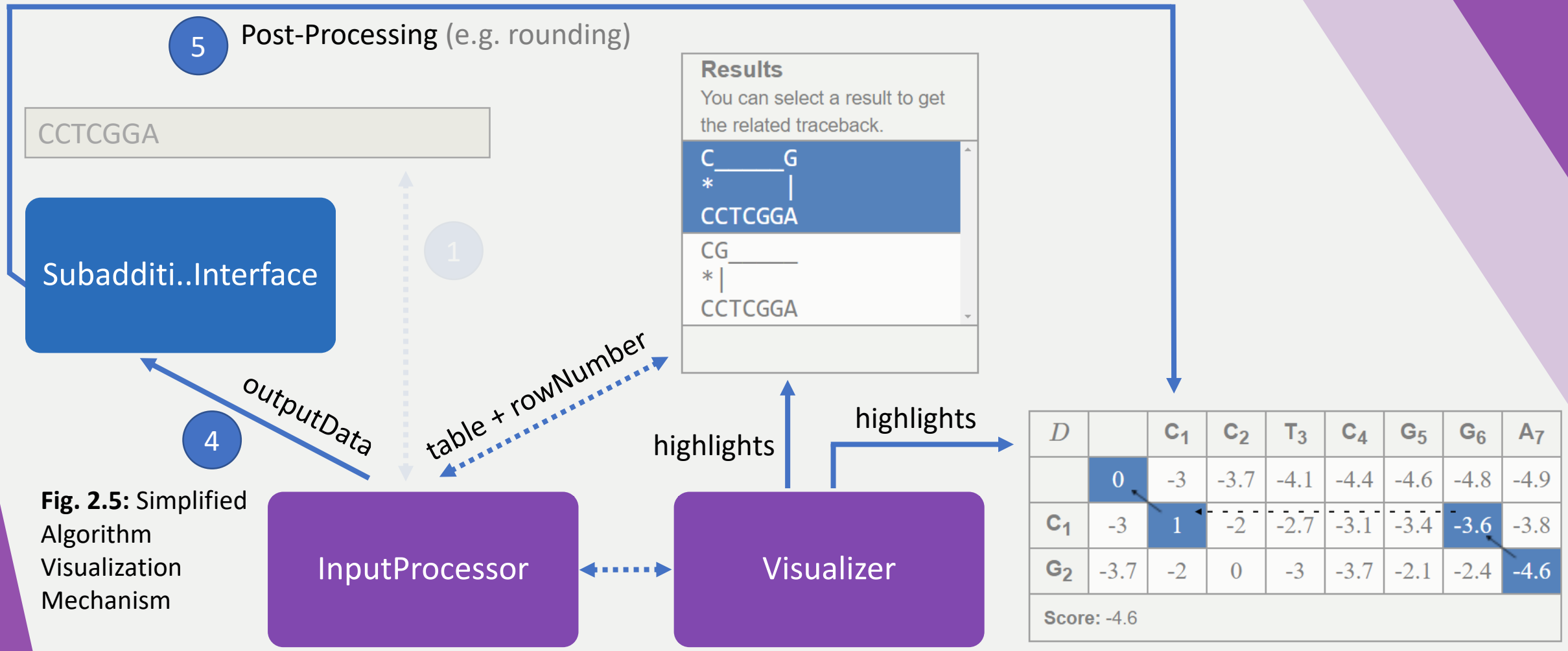


Fig. 2.5: Simplified Algorithm Visualization Mechanism

D		C_1	C_2	T_3	C_4	G_5	G_6	A_7
	0	-3	-3.7	-4.1	-4.4	-4.6	-4.8	-4.9
C_1	-3	1	-2	-2.7	-3.1	-3.4	-3.6	-3.8
G_2	-3.7	-2	0	-3	-3.7	-2.1	-2.4	-4.6
Score: -4.6								

ARCHITECTURE

SEPARATION OF ALGORITHM AND INTERFACE LOGIC



D		C_1	C_2	T_3	C_4	G_5	G_6	A_7
	0	-3	-3.7	-4.1	-4.4	-4.6	-4.8	-4.9
C_1	-3	1	-2	-2.7	-3.1	-3.4	-3.6	-3.8
G_2	-3.7	-2	0	-3	-3.7	-2.1	-2.4	-4.6

Fig. 2.5: Simplified Algorithm Visualization Mechanism

ARCHITECTURE
ADVANTAGES – IDEA BEHIND

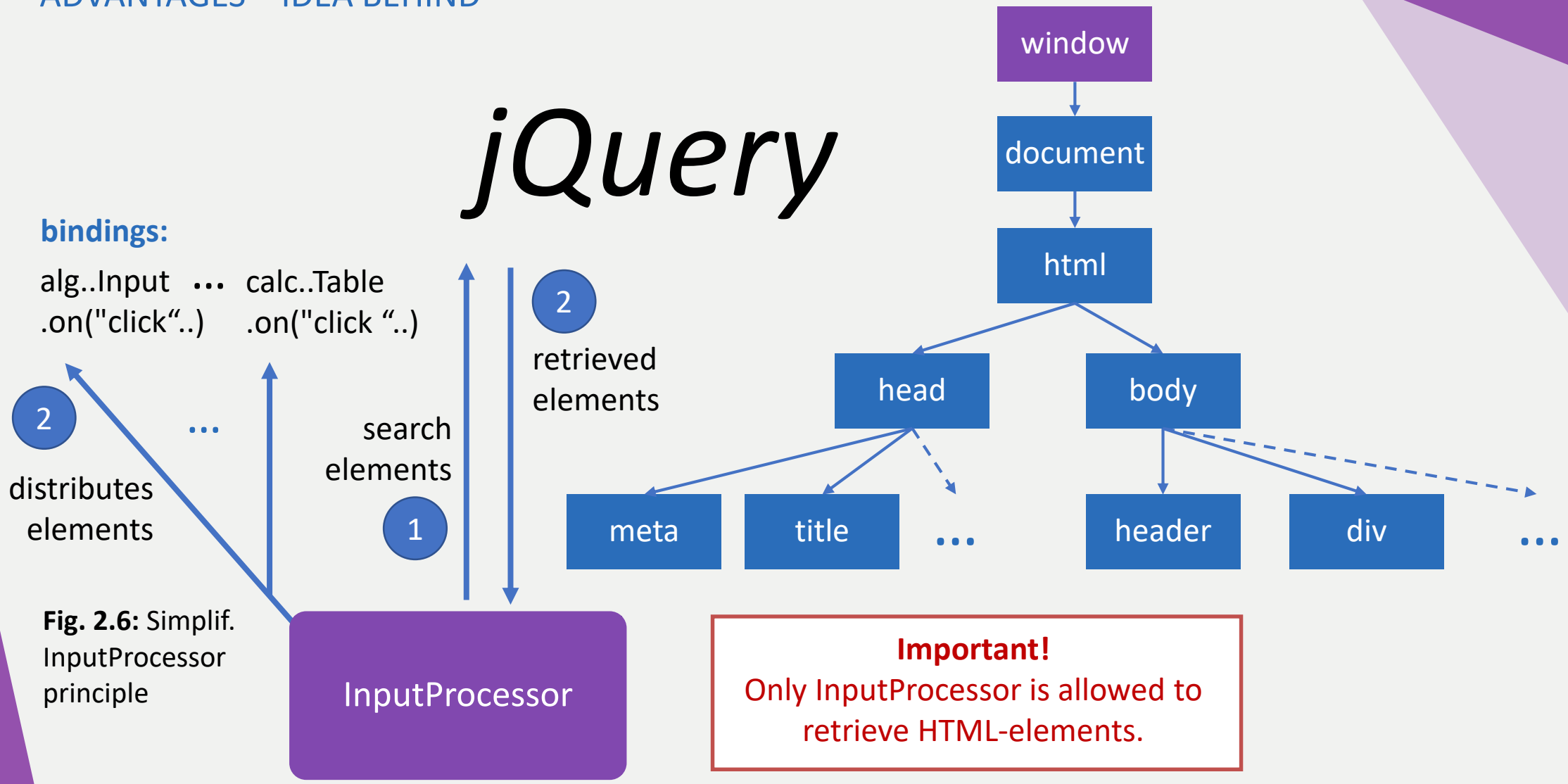


Fig. 2.6: Simplif. InputProcessor principle

Important!
Only InputProcessor is allowed to retrieve HTML-elements.

Sources

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