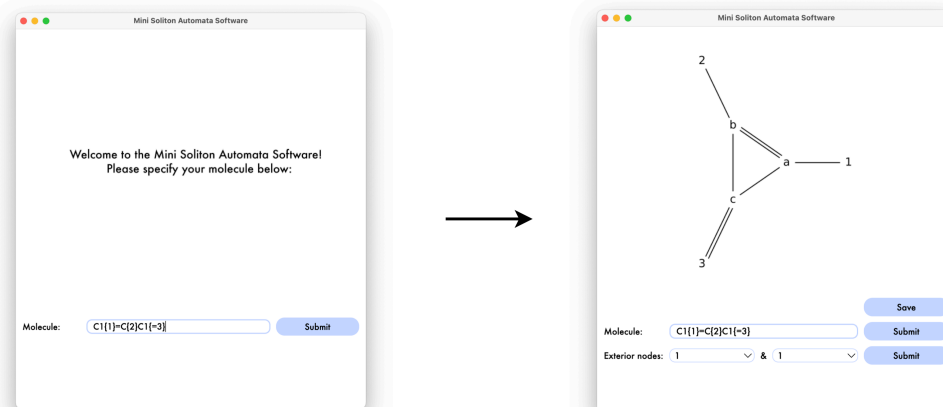


Informationen zur Mini Soliton Automata Software

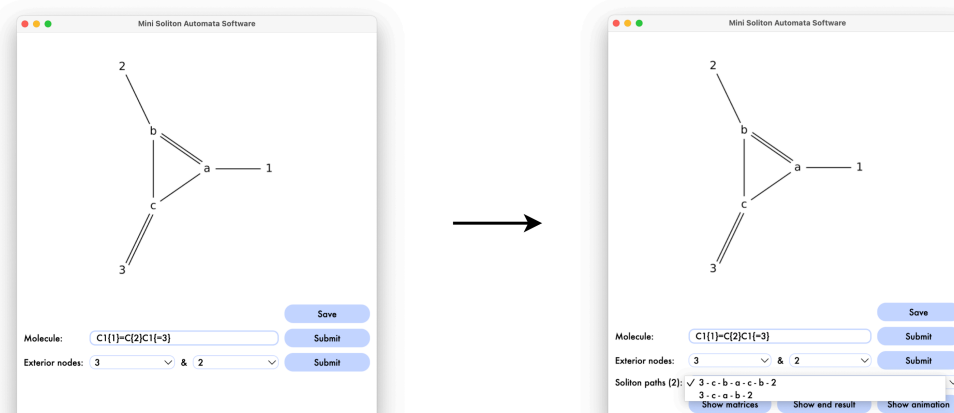
Mithilfe der Mini Soliton Automata Software lassen sich alle Solitonpfade zwischen zwei externen Knoten in einem Solitongraphen berechnen. Sie implementiert somit einen Teil von einem Solitonautomaten. Ihre graphische Benutzeroberfläche ermöglicht verschiedene Eingaben und zeigt Visualisierungen des Solitongraphs und der gefundenen Solitonpfade. Unter <https://github.com/schulz-helena/soliton-implementation> befindet sich das Repository zur Software.

Bedienungsanleitung:

Nach dem Starten der Software ist zunächst im verfügbaren Eingabefeld das Molekül anzugeben, in welchem nach Solitonpfaden gesucht werden soll. Die Spezifizierung eines Moleküls folgt einer bestimmten Syntax (siehe Abschnitt „Input-Syntax für die Spezifizierung eines Moleküls“). Mit „Submit“ kann die Eingabe bestätigt werden. Nun wird im oberen Bereich der Benutzeroberfläche der entsprechende Solitongraph angezeigt. In der Visualisierung sind innere Knoten als Buchstaben und externe Knoten als Zahlen gekennzeichnet. Mit „Save“ kann der Graph als PNG-, JPG- oder JPEG-Datei gespeichert werden.



Als nächstes können aus allen externen Knoten des Solitongraphs zwei externe Knoten ausgewählt werden. Im ersten ausgewählten Knoten betritt das Soliton den Solitongraphen und im zweiten verlässt es ihn wieder. Ist diese Eingabe mit dem „Submit“-Button direkt hinter der Knotenauswahl bestätigt worden, werden alle möglichen Solitonpfade zwischen den beiden externen Knoten berechnet.



Nun kann einer der berechneten Pfade ausgewählt werden. Mit „Show matrices“ öffnet sich ein Fenster mit den Adjazenzmatrizen des Solitongraphs zu jedem Zeitpunkt des Durchlaufens des Solitonpfades. Mit „Save“ kann ein Textdokument heruntergeladen werden, was diese

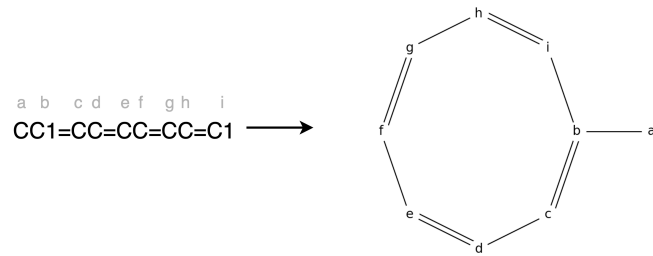
Figure 10 shows three screenshots of the Mini Soliton Automata Software. The first screenshot shows the 'Adjacency Matrices' window, which displays matrices for 'Mole' and 'Ext' and a list of soliton paths. The second screenshot shows the 'End result' window, which displays a graph with nodes 'a', 'b', 'c' and edges '2', '3'. The third screenshot shows the 'Animation' window, which displays the same graph with an animation of the path.

Die Input-Syntax basiert auf dem Simplified Molecular Input Line Entry Specification (SMILES) Strukturcode, mit dem Moleküle als Strings dargestellt werden können. Diese SMILES-Repräsentation wird um eine Regel zum Spezifizieren von externen Knoten erweitert.

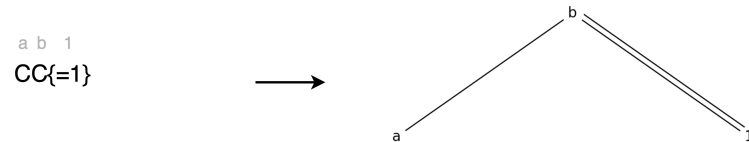
$$a \rightarrow a$$

$\begin{array}{cccccc} a & b & c & d & e & f \\ \text{CC} & = \text{CC} & = \text{CC} & & & \\ \text{C} & - \text{C} & = \text{C} & - \text{C} & = \text{C} & - \text{C} \end{array} \longrightarrow \begin{array}{c} \text{b} \\ \diagup \quad \diagdown \\ \text{a} \quad \text{c} \quad \text{d} \quad \text{e} \\ \diagdown \quad \diagup \quad \diagdown \quad \diagup \\ \text{c} \quad \text{e} \end{array}$

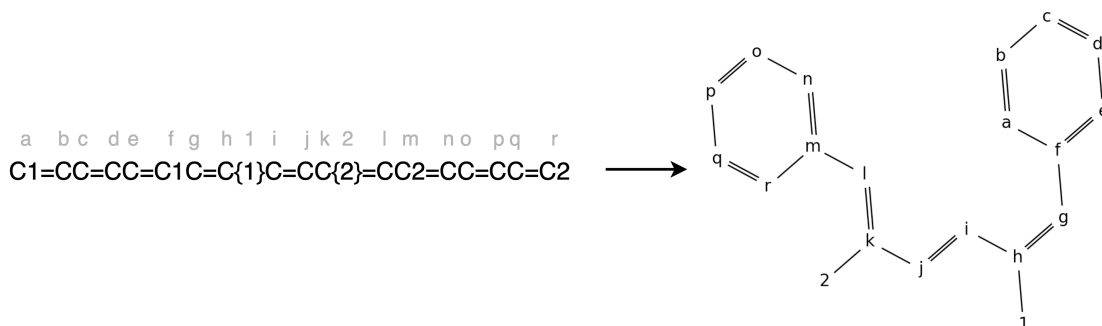
$$\begin{array}{cccccc} a & b & & c & d & e & f \\ \text{CC}(=\text{CC})\text{C}=\text{C} & \longrightarrow & \begin{array}{c} \text{a} \text{---} \text{b} \begin{array}{l} \nearrow \text{c} \text{---} \text{d} \\ \searrow \text{e} \text{---} \text{f} \end{array} \end{array} \end{array}$$



Bei einem Ring werden die Atome an der Stelle, an der der Ring geschlossen wird, mit der gleichen Nummer markiert (z.B. *C1* und *C1*). Zwischen diesen beiden Atomen wird eine Einfachbindung gezogen.
 Externe Knoten werden als eine Bindung und eine Zahl in einer geschweiften Klammer dargestellt.



Wichtig bei der Benutzung von Klammern (*()* bei einer Abzweigung oder *{}* bei einem externen Knoten) ist, dass die Bindung zwischen dem Atom vor der Klammer und dem Atom hinter der öffnenden Klammer immer in die Klammer geschrieben wird (nicht davor).



Beispiele für Solitongraphen:

Implementierung:

Gewählte Bedingung für Pfadberechnung:

Der Algorithmus zur Berechnung der Solitonpfade benutzt die „Vergangenheitsbedingung“. Das heißt, jedes Mal, wenn entschieden wird, welcher Knoten als nächstes betreten werden kann, wird die zuletzt traversierte Kante betrachtet. Nur wenn diese ein anderes Gewicht hat, als die Kante zu einem möglichen nächsten Knoten, kann dieser Knoten ausgewählt werden.

Benutzte Frameworks:

- *rdkit*: Wandelt den SMILES-String in eine Repräsentation eines Moleküls um. Dieses Molekül fungiert als eine Art Zwischendarstellung und wird später in einen Graphen umgewandelt.

Berechnet Positionen für Atome und Bindungen, die einer chemischen Darstellung eines Moleküls entsprechen, weshalb diese Positionen auch für die Darstellung als Solitongraph genutzt werden.

- *pysmiles*: Berechnet die exakten Bindungstypen jeder Bindung.
- *networkx*: Plottet den Graphen.
- *matplotlib*: Plottet auf die *networkx*-Darstellung des Graphs für jede Doppelbindung eine zweite Kante und einen schwarzen Punkt als Repräsentation des Solitons.
- *numpy*: Realisiert die Adjazenzmatrizen des Solitongraphs.
- *Pillow*: Wandelt Visualisierungen und Animationen des Solitongraphs in Images um und gibt damit die Möglichkeit, diese im memory zu halten, statt sie lokal abzuspeichern.
- *PyQt5*: Realisiert die graphische Benutzeroberfläche.

Installation und Starten der Software:

Die Mini Soliton Automata Software kann mittels pip von GitHub heruntergeladen werden. Deshalb muss zunächst auf dem Zielrechner, bzw. der Zielumgebung pip installiert sein.

Mit dem Kommandozeilenbefehl

```
pip install git+https://github.com/schulz-helena/soliton-implementation
```

wird die Software als Python Package heruntergeladen.

Nun kann sie mittels

```
mini-soliton-automata-software
```

gestartet werden.