Chemisches Digitalcomputermodell "Münzenrückgabe"

Projektarbeit

Friedrich-Schiller-Universität Jena
Fakultät für Mathematik und Informatik
Molekulare Algorithmen

Fionn Daire Keogh
Carsten Scholl

betreut von
PD Dr-Ing. habil. Thomas Hinze

Jena, 18. Juli 2022

Zusammenfassung

In dieser Arbeit stellen die Autoren ein chemisches Digitalcomputermodell vor, welches in der Lage ist, die Münzrückgabe im Bereich zwischen 0 und 99 Cent zu berechnen. Es wird die Funktionsweise erläutert und einige Simulationsbeispiele gezeigt. Zusätzlich wird ein chemisches Analogcomputermodell vorgestellt, welches in der Lage ist die selbe Aufgabe zu lösen und mit dem digitalen Modell verglichen.

1 Einführung

1.1 Chemische Analogcomputer

Analogcomputer waren die Vorläufer der digitalen Computer. Dabei ist die Berechnung relativ dicht an das zugrunde liegende Funktionsprinzip gekoppelt. Bei chemischen Analogcomputern werden die Berechnungen durch chemische Reaktionen dargestellt, wobei jede Spezies eine Variable und ihre Stoffmengenkonzentration die zugeordnete Belegung repräsentiert. Die Eingabe erfolgt dann über die initiale Stoffmengenkonzentration der Reaktanten. So lassen sich die mathematischen Grundrechenarten wie Addition, Subtraktion, Multiplikation oder Division als chemisches Reaktionsnetzwerk darstellen[9].

Im folgenden Beispiel soll die nicht-negative Subtraktion in einem chemischen Analogcomputer verdeutlicht werden. Dazu benötigt man vier Reaktionen, einen Minuenden X1, einen Subtrahenden X2, die Differenz Y, sowie die Hilfsspezies Z.

$$
\begin{aligned}
X_1 &\xrightarrow{k_1} X_1 + Y \\
X_2 &\xrightarrow{k_1} X_2 + Z \\
Y &\xrightarrow{k_1} \emptyset \\
Y + Z &\xrightarrow{k_2} \emptyset
\end{aligned}
$$

Aus diesem Reaktionsnetzwerk ergeben sich folgende Differenzialgleichungen:

$$
\begin{aligned}
[\dot{X}_1] &= 0 \\
[\dot{X}_2] &= 0 \\
[\dot{Y}] &= -k_2[Y][Z] - k_1[Y] + k_1[X_1] \\
[\dot{Z}] &= k_1[X_2] - k_2[Y][Z]
\end{aligned}
$$

Um nun eine konkrete Rechnung durchzuführen, werden die Ausgangskonzentrationen von X1 und X2 auf die gewünschten Werte gesetzt, Y auf 0. Nachdem das System einen stationären Zustand erreicht hat, kann das Ergebnis der nicht-negativen Subtraktion an der Stoffmengenkonzentration von Y abgelesen werden.

1.2 Chemische Digitalcomputer

Digitalcomputer haben einige Vorteile gegenüber ihren Vorgängern, die dafür gesorgt haben, dass sie diese heutzutage fast vollständig abgelöst haben. Dazu gehören unter anderem eine niedrigere Anfälligkeit für Störungen sowie höhere Genauigkeit und Geschwindigkeit. Chemische Digitalcomputermodelle arbeiten im Gegensatz zu Analogcomputern

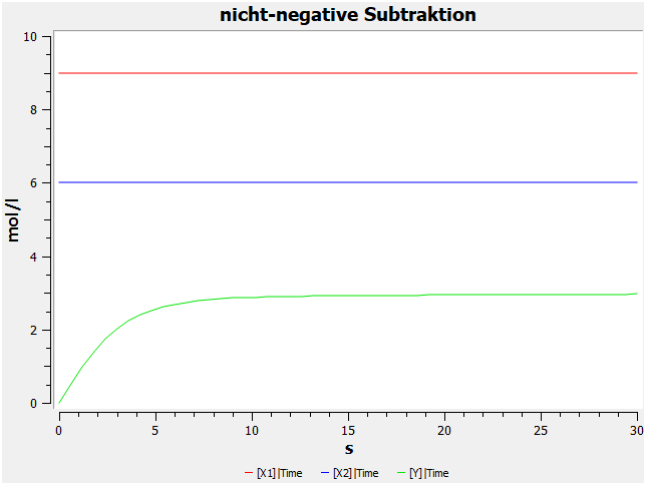

Abbildung 1: Subtraktion im chemischen Analogcomputer am Beispiel 9 - 6 = 3

im diskreten Wertebereich und benötigen einen Taktgeber, um die diskreten Rechenschritte voneinander zu trennen. Statt Berechnungen im kontinuierlichen Wertebereich durchzuführen, verwenden Digitalcomputer einfache Logikoperationen[9].

X1	X2	Y
0	0	1
0	1	1
1	0	1
1	1	0

Tabelle 1: Beispiel einer Übergangstabelle für NAND-Gatter

1.3 Aufgabenstellung

Ziel dieser Arbeit war es, ein chemisches Digitalcomputermodell zu entwickeln, welches für einen Cent-Betrag zwischen 0 und 99 die minimale Zusammensetzung von Münzen berechnet, um diesen Betrag zu bilden. Dabei stand es frei, die Eingabe entweder 10-wertig oder binär erfolgen zu lassen. Die Ausgabe der Anzahl der jeweiligen Münzen sollte in ternärer Logik erfolgen, also entweder als 0, 1 oder 2. Die Berechnung sollte durch geeignete Logikfunktionen erfolgen und möglichst optimiert werden. Im Anschluss sollten mehrere Fallbeispiele gezeigt und erläutert werden.

2 Methoden

In diesem Abschnitt soll das konstruierte Modell des chemischen Digitalcomputers beschrieben werden und die zur Realisierung verwendete Software.

2.1 Naiver Digitalcomputer zur Lösung der Aufgabe

Um die in Tabelle 2 beschriebene Ausgabe an Münzen zu erzielen, kann ein chemischer Digitalcomputer mit einem naiven Ansatz zur Lösung realisiert werden.

x	1 Cent	2 Cent	5 Cent	10 Cent	20 Cent	50 Cent
0	0	0	0	0	0	0
1	1	0	0	0	0	0
2	0	1	0	0	0	0
3	1	1	0	0	0	0
⋮						
58	1	1	1	0	0	1
⋮						
99	0	2	1	0	2	1

Tabelle 2: Tabelle der gewünschten Ausgabewerte

Dieser lässt sich schnell in Worten beschreiben. Für jeden korrekten Eingangs-Zustand wird für jede Münze definiert, ob sie vorhanden ist oder nicht.
Für jede Münze werden also 100 logische Anweisungen definiert. So werden dann 600 Anweisungen benötigt, um jede Münz-Ausgabe zu beschreiben. Diese können dann mit chemischen Reaktionen realisiert werden, um das Digitalcomputermodell chemisch zu übersetzen (siehe hierzu NAND-Gatter). Tabelle 3 zeigt einige logische Anweisungen.

dec	I64	I32	I16	I8	I4	I2	I1	5 Cent
0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	1	0
2	0	0	0	0	0	1	0	0
3	0	0	0	0	0	1	1	0
4	0	0	0	0	1	0	0	0
5	0	0	0	0	1	0	1	1
				⋮				
98	0	0	0	0	1	0	1	1
99	0	0	0	0	1	0	1	1

Tabelle 3: Beispiele logischer Anweisungen um zu prüfen, ob eine 5 Cent Münze ausgegeben werden soll. Jedes Eingangs-Bit ist mit *und* verknüpft ($\wedge$).

Wie deutlich zu erkennen ist, kann man einige Anweisungen zusammenfassen. Im nächsten Abschnitt werden einige Optimierungen beschrieben und das Modell des Computers erläutert.

2.2 Modell des chemischen Digitalcomputers

Das Herzstück des chemischen Digitalcomputers ist der Taktgeber. Aus der Literatur sind einige chemische Oszillatoren bekannt auf verschiedenen Basis-Eigenschaften. Beispiele beinhalten unter anderem die Lotka-Volterra Gleichungen[2], den Repressilator nach Elowitz und Leibler[3], den modifizierten Goodwin-Oszillator[7] und den Brüsselator nach Prigogine und Lefever.
Die Autoren dieser Arbeit haben sich für eine nach Faßler *et al.*[8] modifizierte und erweiterte Version des Brüsselators entschieden. Eine Reihe von Kaskaden-Reaktionen der Erhaltungsreaktionen durch simulierte Kippschalter.

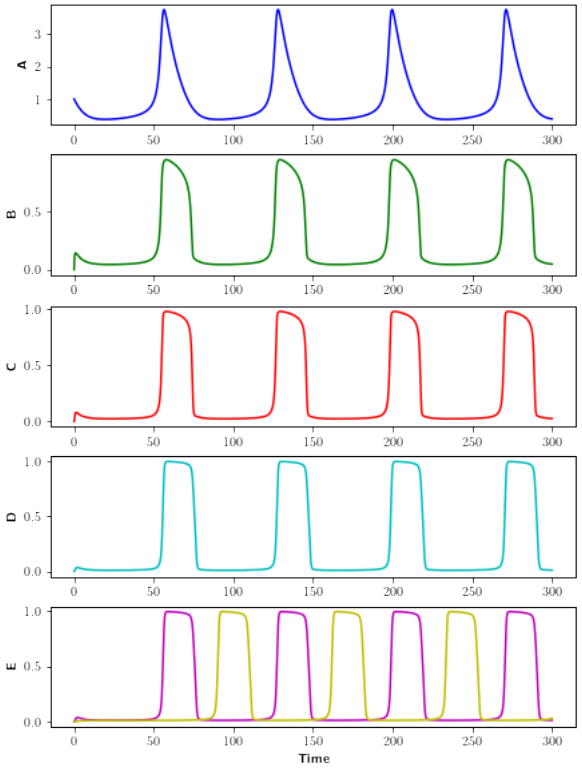

Abbildung 2: Nicht-überlappender binärer Taktgeber nach Faßler *et al.*. **A** zeigt den ursprünglichen Brüsselator. **B-D** zeigen den Stand der Taktgebenden-Spezies nach jeweils einer weiteren Kaskade. Auffällig ist, dass mit jeder weiteren Kaskade der Zerfallsprozess verzögert und gleichzeitig beschleunigt wird. **E** zeigt nun zwei, zeitversetzte, drei Mal kaskadierte Brüsselatoren, die nicht überlappen und binär vorhanden sind.

Die Eingabe des Computers besteht aus sieben Spezies, $I64$, $I32$, $I16$, $I8$, $I4$, $I2$, $I1$. Diese haben eine Anfangskonzentration von 0 mol/l und müssen vom Nutzer manuell gesetzt werden. Akzeptierte Eingaben sind 0 und 1. Mit diesen Spezies lässt sich die binäre Eingabe der Zahlen 0-99 realisieren.
Als Ausgabe erhält der Nutzer sechs Konzentrationen von Spezies ($1er\_Coin$, $2er\_Coin$, $5er\_Coin$, $10er\_Coin$, $20er\_Coin$, $50er\_Coin$), die zu den einzelnen Cent-Münzen korrespondieren. Ausgaben können die Form 0, 1 und im Falle der 2-Cent und 20-Cent Münzen auch 2 haben.
Vor Beginn des ersten Takts, Beginn in Abbildung 2 bei etwa 50 Zeit-Schritten, werden die Eingaben vorbereitet. Das bedeutet, um binäre, logische Abfragen zu realisieren, wird für jede Eingabe, die nicht vorhanden ist (Anfangskonzentration ist nicht auf *1* gesetzt) wird eine *nicht*-Eingabe erstellt, bzw. nicht zerstört.
Dies wird umgesetzt, indem es einen Anfangsbestand von Spezies gibt, die jeweils von der korrespondierenden Eingangs-Spezies zerstört wird (siehe Tabelle 4).
Wenn man nun die Muster der Tabelle 3 betrachtet, fällt auf, dass einige Bits nach Betrachtung einiger vorheriger Bits nicht mehr relevant sind. Beispielsweise sind bei der 50-Cent Münze und einem Vorhandensein von $I64$ alle anderen Bits/Spezies egal, die 50-Cent Münze ist bei jeder Ausgabe mit dabei und kann schon nach nur einer Bit-Betrachtung auf 1 gesetzt werden.

Mit jedem Takt wird nun eine Bit-Position betrachtet und entschieden, ob bei Vorhan-

$$I64 + A64 \rightarrow I64$$
$$I32 + A32 \rightarrow I32$$
$$I16 + A16 \rightarrow I16$$
$$I8 + A8 \rightarrow I8$$
$$I4 + A4 \rightarrow I4$$
$$I2 + A2 \rightarrow I2$$
$$I1 + A1 \rightarrow I1$$

Tabelle 4: Wenn nun eine Eingabe-Spezies ($I...$) vorhanden ist, wird ihr Komplement ($A...$) zerstört, sollte die Eingabe nicht vorhanden sein, bleibt das Komplement vorhanden.

densein gewisser Spezies, oder Abwesenheit, bzw. Vorhandensein der Komplemente eben dieser, eine Entscheidung über eine Münze getroffen werden kann. So ist nach maximal sieben Takten der Entscheidungs-Prozess beendet.

In einem achten Takt wird aus zwei verschiedenen 2-Cent ($2.1er\_Coin$ und $2.2er\_Coin$) und 20-Cent Münzen (analog) eine Münze ($2er\_Coin$ und $20er\_Coin$) im Ternärsystem erstellt. Nach diesem Schritt, bei etwa 350 Zeit-Schritten in der mitgelieferten Simulation können die Konzentrationen der Münzen ausgelesen werden.
In Abschnitt 3 werden beispielhaft Simulationen und Ergebnisse präsentiert.

2.3 Software

Für die Umsetzung des Modells wurde **CopasiUI 4.36** (Build 260)[5] genutzt. Das erstelle Copasi-File ist dem Supplement-Material beigefügt.

Es wurden auch einige Methoden zur erleichterten Bedienung und zur Visualisierung erzeugt. Diese sind in einem Jupyter Notebook dem Supplement-Material beigefügt.
Diese Python-Skripte nutzen die Bibliotheken, NumPy[4], Matplotlib[6] und basico[1].

```python
def run_for_input(_input, start=0, end=350, steps=6000):
    binary = bin(_input)[2:]
    inp = [0,0,0,0,0,0,0]
    for i in range(len(binary)):
        j = len(inp)-(i+1)
        k = len(binary)-(i+1)
        inp[j] = int(binary[k])

    set_species("I_64", initial_concentration=inp[0]) # basico Methode
    set_species("I_32", initial_concentration=inp[1])
    set_species("I_16", initial_concentration=inp[2])
    set_species("I_8", initial_concentration=inp[3])
    set_species("I_4", initial_concentration=inp[4])
    set_species("I_2", initial_concentration=inp[5])
    set_species("I_1", initial_concentration=inp[6])
    tc = run_time_course(start, end, steps) # basico Methode
    return tc
```

Code Listing 1: Python Methode, welche einen `_input` als Integer nimmt und diesen in eine Binärzahl umwandelt. Diese wird anschließend genutzt um die Initialkonzentrationen zu setzen. Der Zeitverlauf des Models wird dann simuliert und als *DataFrame* `tc` zurück gegeben

```python
def plot_split(_input, observables = ["1er_Coin", "2er_Coin", "5er_Coin", "10er_Coin", "20er_Coin", "50er_Coin"], names = ["1 Cent", "2 Cent", "5 Cent", "10 Cent", "20 Cent", "50 Cent"]):
    colors = ['b', 'g', 'r', 'c', 'm', 'y']
    df = run_for_input(_input)
    fig, axes = plt.subplots(ncols = 1, nrows = len(observables), constrained_layout=True, figsize=(6, 8))
    for i in range(len(observables)):
        ax = axes[i]
        ax.plot(df[observables[i]], label=str(names[i]), color=colors[i])
        ax.set_ylim([-0.1, 2.1])
        if i == len(observables)-1:
            ax.set_xlabel("time")
            ax.set_ylabel("# Coins")
        labelLines(ax.get_lines(), zorder=2.5)
    fig.show()
```

Code Listing 2: `plot_split()` bekommt als Eingabe einen Integer. Dieser wird genutzt um den Zeitverlauf zu berechnen. Für jede der relevanten Ausgaben wird ein Graph erstellt (siehe als Beispiel Figure 5 für die Ausgabe von 42 Cent).

3 Simulationsstudien

Zur Verdeutlichung der Arbeitsweise und Korrektheit des chemischen Digitalcomputers wurden einige Simulationen durchgeführt.

3.1 Testen der Korrektheit des Models

Da die möglichen Eingaben für das Model sehr begrenzt sind, war es einfach möglich, eine Test-Funktion zu schreiben, welche alle möglichen Eingaben auf Korrektheit prüft. Listing 3 zeigt die Test-Funktion. Dazu wird für jeden möglichen Input die `run_for_input`-Funktion aufgerufen und die letzten Konzentrationen der Cent-Münzen mit den Beträgen der Münzen multipliziert und dann zusammen summiert. Ist diese Summe gleich dem Input, ist die richtige Menge Münzen ausgegeben worden.

3.2 Ausgewählte Simulationen

Im Folgenden sind einige ausgewählte Zahlen-Beispiele für Simulationen. Jede der Visualisierungen wurde mit der oben beschriebenen Funktion erstellt.

0

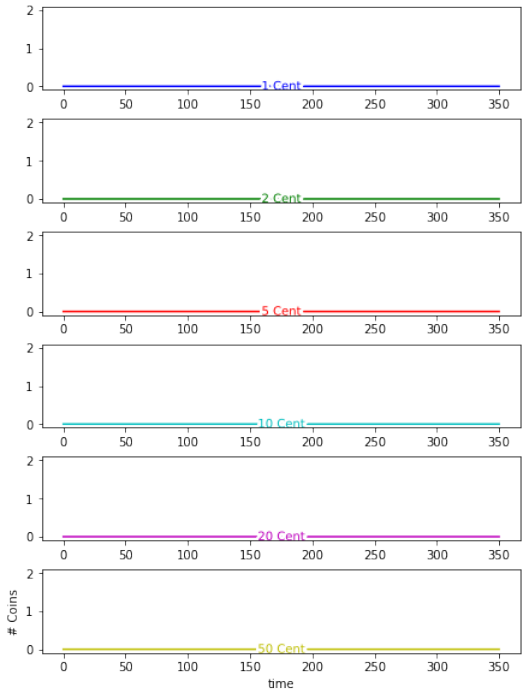

Abbildung 3: Das triviale Beispiel der 0 ergibt für die Eingabe $[0, 0, 0, 0, 0, 0, 0]$ die Ausgabe $[0, 0, 0, 0, 0, 0]$. Keine der Reaktionen wurde geschaltet, dementsprechend wurde keine der Münz-Konzentrationen erhöht.

7

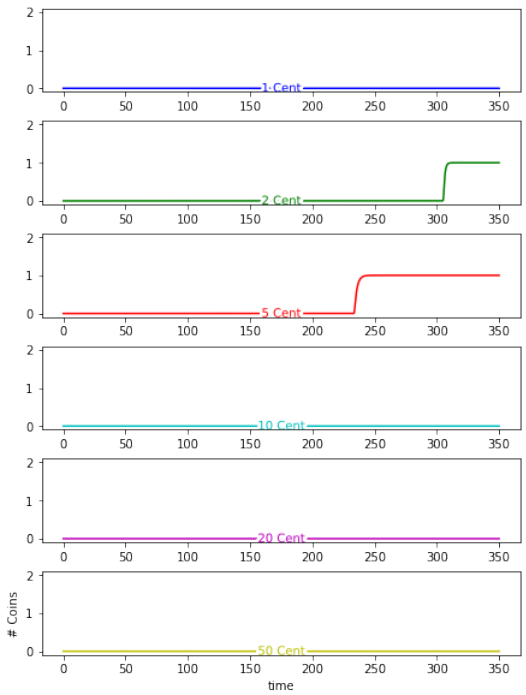

Abbildung 4: Als einstelliges Beispiel wurde die 7 gewählt. Mit der resultierenden binären Eingangskonzentrationen $[0, 0, 0, 0, 1, 1, 1]$ wurden die Reaktionen $000011 \rightarrow 5er$-Münze und $0000111 \rightarrow 2er$-Münze aktiviert. Die 5 wurde in einem früheren Takt als die 2 aktiviert, da nicht alle Bits angeschaut werden müssen.

42

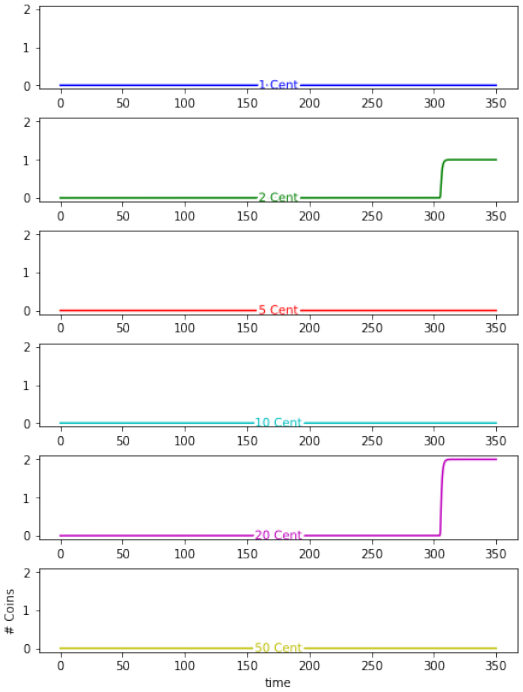

Abbildung 5: Das Beispiel der 42 ergibt für die Eingabe $[0, 1, 0, 1, 0, 1, 0]$ die Ausgabe $[0, 1, 0, 0, 2, 0]$. Hier ist besonders hervorzuheben, dass gut sichtbar im letzten Schritt die 20 Cent Kurve auf 2 erhöht.

88

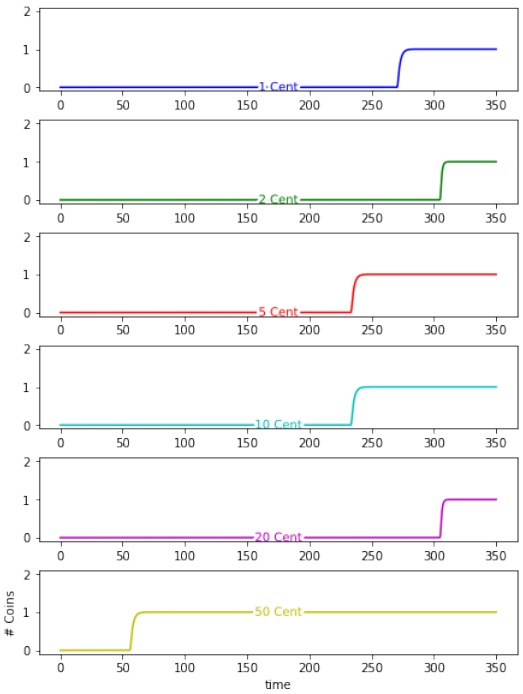

Abbildung 6: Die 88 wurde als Eingabe gewählt, da hier alle Münzen einmal ausgegeben werden. Die Eingabe $[1, 0, 1, 1, 0, 0, 0]$ ergibt die Ausgabe $[1, 1, 1, 1, 1, 1]$.

4 Konklusion und Diskussion

4.1 Vergleich mit Analogcomputermodell

4.1.1 Funktionsweise

Alternativ lässt sich das Münzrückgabeproblem auch durch einen chemischen Analogcomputer lösen. Anstatt Logikverschaltungen zu verwenden, werden die Berechnungen mithilfe einfacher Rechenoperationen durchgeführt. Die Funktionsweise ist ähnlich, der, wie ein Mensch vorgehen würde, um die Aufgabenstellung zu lösen. Beginnend mit der größten Münze wird geprüft, ob sie in den Eingabebetrag passt. Wenn ja, wird der Wert der Münze vom Betrag abgezogen und die nächstkleinere Münze überprüft, bis alle Münzen abgehandelt wurden und der Restbetrag 0 Cent beträgt.

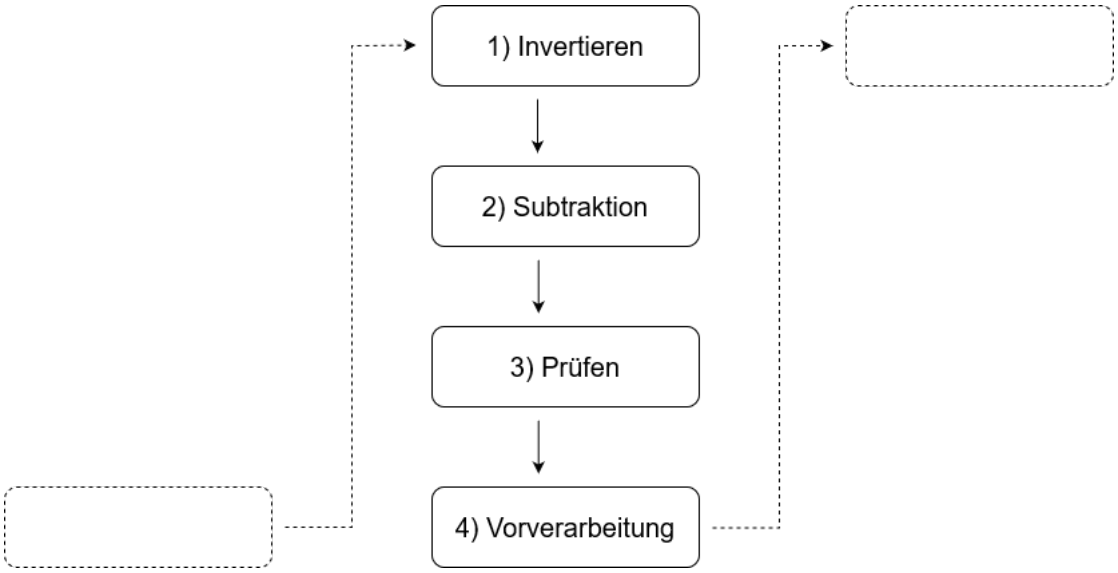

Abbildung 7: Schematische Darstellung der Berechnungsschritte

Anders als bei dem zuvor beschriebenen chemischen Analogcomputer erfolgt die Eingabe des Geldbetrages hier nicht in Binärform, sondern 10-wertig. Die initiale Stoffmengenkonzentration der Input-Spezies erhält also einen Wert zwischen 0 und 99 mol/l. Für jede Münzklasse in absteigender Reihenfolge durchläuft der Computer dann die in Abbildung 7 gezeigten Schritte. Um diskrete, voneinander getrennte Rechenschritte durchzuführen, wurde auch hier ein nicht-überlappender binärer Taktgeber (siehe Abbildung 2) verwendet.

Schritt 1) Zuerst wird der Eingabewert mit einem für jede Münzklasse festgelegten Wert invertiert. So wird beispielsweise bei den 5 Cent Münzen der Eingabewert von 10 abgezogen. Dazu wird die in Abbildung 1 gezeigte nicht-negative Subtraktion verwendet. Die Invertierung dient dem Zweck, dass nach der Subtraktion in Schritt 2) die Fälle, in denen die Münze in den Eingabebetrag passt und die, in denen sie nicht passt, unterscheidbar sind.

Münze	Invertierung mit
50c	100
20c	40
10c	20
5c	10
2c	4
1c	2

Tabelle 5: Werte mit denen für jede Münze invertiert wird

Schritt 2) In diesem Schritt wird von dem invertierten Ergebnis aus Schritt 1) der Wert der jeweiligen Münze abgezogen. Dies erfolgt ebenfalls durch nicht-negative Subtraktion. Passt die Münze in den Eingabebetrag, ist das Ergebnis der Subtraktion null, passt es nicht, ist es größer als null. Ohne eine Invertierung in Schritt 1) wäre der Fall, dass die Münze genau in den Eingabewert passt, nicht von dem Fall zu unterscheiden, dass sie nicht passt. In beiden Fällen wäre das Ergebnis der Subtraktion null.

Schritt 3) Um nun zu prüfen, ob die Münze vorhanden sein soll oder nicht, findet nun eine Reaktion statt, in der die Spezies, welche das Ergebnis der Subtraktion aus Schritt 2) darstellt, die Spezies, welche die entsprechende Münze darstellt, auffrisst. Die initialen Stoffmengenkonzentrationen der Münz-Spezies betragen 1, bzw. 0 für die 20 Cent und 2 Cent Münzen. In der folgenden Beispielreaktion bestimmen $C2$ und $T15$ den Takt, $5er\_Sub5\_Y$ ist die Ergebnisspezies der Subtraktion aus Schritt 2), welche, wenn sie vorhanden ist die $5er\_Coin$ Spezies abbaut und $Anti\_5er\_Coin$ erzeugt.

$$5er\_Coin + 3 \cdot C2 + 3 \cdot T15 + 3 \cdot 5er\_Sub5\_Y \rightarrow$$
$$3 \cdot C2 + 3\dot{T}15 + 3 \cdot 5er\_Sub5\_Y + Anti\_5er\_Coin$$

Schritt 4) Im letzten Schritt muss noch die Eingabe für die nächste Münze vorbereitet werden. Dazu erfolgt zunächst eine Multiplikation der Stoffmengenkonzentration der aktuellen Münze mit ihrem Wert. Dann wird erneut eine Subtraktion durchgeführt, in der das Ergebnis dieser Multiplikation von der Eingabe der aktuellen Münze abgezogen wird. Das Ergebnis dieser Subtraktion kann dann als Eingabe für die nächste Münze verwendet werden.

Die Berechnung für die 2 Cent und 20 Cent Münzen bilden eine Ausnahme. Hier sind die Anfangskonzentrationen der Münz-Spezies gleich null. Dann wird wie oben beschrieben invertiert und zuerst für die erste der beiden möglichen Münzen geprüft, ob sie passt. Wenn ja, wird im Anschluss die zweite Münze überprüft. Dann werden entsprechend viele Münz-Spezies erzeugt und die Eingabe für die nächste Münze vorbereitet, wie in Schritt 4) beschrieben.

4.1.2 Simulationsbeispiele

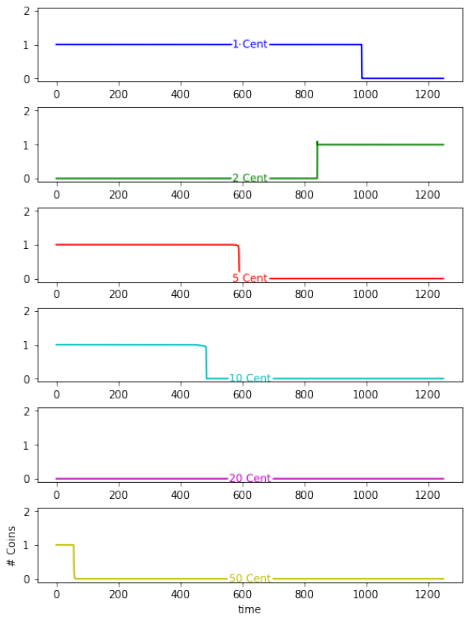

Abbildung 8: Beispiel mit INPUT=2. Man kann erkennen, dass die Münzen in absteigender Reihenfolge berechnet werden. Für die Eingabe 2 ist die Ausgabe nur eine 2 Cent Münze.

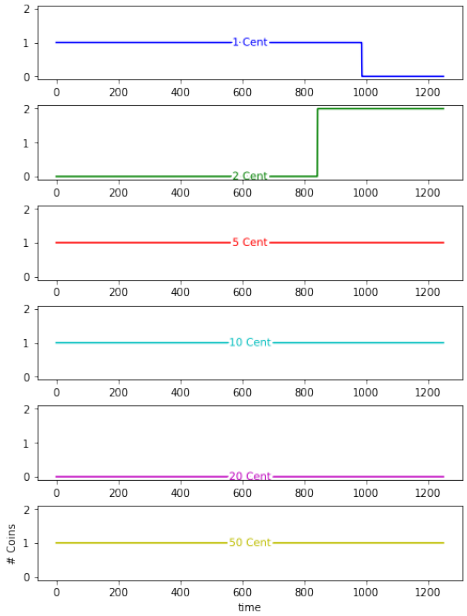

Abbildung 9: Beispiel mit INPUT=69. Hie ist die Ausgabe eine 50 Cent Münze, eine 10 Cent Münze, eine 5 Cent Münze und zwei 2 Cent Münzen.

4.1.3 Vergleich

Ein deutlich erkennbarer Vorteil des digitalen Computermodells ist die Laufzeit. Das unter 2.2 beschriebene Modell benötigt knapp über 300 Sekunden, während der Analogcomputer ca. 1.000 Sekunden braucht. Das liegt daran, dass die Implementierung des Analogcomputers 27 Takte umfasst, die des Digitalcomputers aber nur 8 Takte, was sich proportional in der Geschwindigkeit bemerkbar macht. Was den Umfang der beiden Modelle angeht, liegen sie sehr dicht beieinander, was die Anzahl der Reaktionen angeht, wobei das Digitalcomputermodell deutlich weniger Spezies benötigt. Das analoge Computermodell hat den Vorteil, dass es leichter verständlich ist, da die Berechnung näher an der intuitiven Herangehensweise liegt.

	digital	analog
# Spezies	53	131
# Reaktionen	204	206
# Takte	8	27

Tabelle 6: Vergleich der Komplexität zwischen chemischem Analog- und Digitalcomputer

4.2 Konklusion

In dieser Arbeit haben die Autoren gezeigt, wie ein chemischer Digitalcomputer zur Berechnung der Münzrückgabe umgesetzt werden kann. Von der Erklärung eines naiven Ansatzes wurde zu einem effektiveren Modell geführt. Es wurde aufgeführt, dass nicht immer alle Bits notwendig sind, um eine Entscheidung über eine Münze treffen zu können.
Als Beispiel wurden die Visualisierungen verschiedener Simulationen beschrieben.
Zuletzt wurde auch ein analoges Computermodell zum Lösen desselben Problems vorgestellt und mit der Funktionsweise des Digitalcomputers verglichen. Es wurde darauf

geschlossen, dass das Digitalmodell schneller und genauer arbeitet als das Analoge.

Weiterführend könnte man das Modell noch erweitern, damit sowohl eine Eingabe als Dezimal- aber auch als Binärzahl möglich wäre.

Literatur

[1] Frank Bergmann und lilijap. *copasi/basico: Release 0.7*. Version v0.7. Nov. 2021. DOI: 10.5281/zenodo.5723018. URL: https://doi.org/10.5281/zenodo.5723018.

[2] "Elements of Physical Biology". In: *Nature* 116.2917 (1925), S. 461–461. DOI: 10.1038/116461b0. URL: https://doi.org/10.1038/116461b0.

[3] Michael B. Elowitz und Stanislas Leibler. "A synthetic oscillatory network of transcriptional regulators". In: *Nature* 403.6767 (2000), S. 335–338. DOI: 10.1038/35002125. URL: https://doi.org/10.1038/35002125.

[4] Charles R. Harris u. a. "Array programming with NumPy". In: *Nature* 585.7825 (Sep. 2020), S. 357–362. DOI: 10.1038/s41586-020-2649-2. URL: https://doi.org/10.1038/s41586-020-2649-2.

[5] Stefan Hoops u. a. "COPASI—a COmplex PAthway SImulator". In: *Bioinformatics* 22.24 (Okt. 2006), S. 3067–3074. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btl485. eprint: https://academic.oup.com/bioinformatics/article-pdf/22/24/3067/546592/btl485.pdf. URL: https://doi.org/10.1093/bioinformatics/btl485.

[6] J. D. Hunter. "Matplotlib: A 2D graphics environment". In: *Computing in Science & Engineering* 9.3 (2007), S. 90–95. DOI: 10.1109/MCSE.2007.55.

[7] Oliver Purcell u. a. "A comparative analysis of synthetic genetic oscillators." eng. In: *J R Soc Interface* 7.52 (2010), S. 1503–1524. ISSN: 1742-5662 (Electronic); 1742-5689 (Print); 1742-5662 (Linking). DOI: 10.1098/rsif.2010.0183.

[8] R. Faßler, T. Hinze, T. Lenser, P. Dittrich. "Construction of Oscillating Chemical Register Machines on Binary Numbers using Mass-Action Kinetics". In: (Mai 2008). Friedrich-Schiller-Universita t Jena, Bio Systems Analysis Group.

[9] T Hinze. "Molekulare Algorithmen". Vorlesungsnotizen. 2022.

Appendix

```
def plot_split(_input, observables = ["1er_Coin", "2er_Coin", "5er_Coin", "10er_Coin", "20er_Coin", "50er_Coin"], names = ["1 Cent", "2 Cent", "5 Cent", "10 Cent", "20 Cent", "50 Cent"]):
    colors = ['b', 'g', 'r', 'c', 'm', 'y']
    df = run_for_input(_input)
    fig, axes = plt.subplots(ncols = 1, nrows = len(observables), constrained_layout=True, figsize=(6, 8))
    for i in range(len(observables)):
        ax = axes[i]
        ax.plot(df[observables[i]], label=str(names[i]), color=colors[i])
        ax.set_ylim([-0.1, 2.1])
        if i == len(observables)-1:
            ax.set_xlabel("time")
            ax.set_ylabel("# Coins")
        labelLines(ax.get_lines(), zorder=2.5)
    fig.show()
```

Code Listing 3: Test-Methode für alle möglichen Eingaben. Terminiert diese ohne Konsolen-Meldung war der Test erfolgreich.