Aufgabe 4: Streichholzrätsel

Team-ID: 01011

Team: Jim Maar

Bearbeiter/-innen dieser Aufgabe:
Jim Maar

21. November 2020

Inhaltsverzeichnis

Lösungsidee

Der erste Part der Aufgabe besteht darin, ein geeignetes Format zur Darstellung solcher Streichholzanordnungen zu definieren. Wir stellen die Streichhölzer in einem Koordinatensystem dar. Wenn wir alle möglichen, an einen Punkt angrenzenden Streichhölzer mit der Länge 1 aufzeichnen, stellen wir fest, dass alle Enden der Streichhölzer entweder sin(0) = 0 auf der einen und cos(0) = 1 auf der anderen oder sin(30) = 0.5 auf der einen und cos(30) auf der anderen Achse vom Ursprungspunkt entfernt liegen. Wiederholen wir das gleiche an den Enden der Streichhölzer, kommen wir zu dem gleichen Ergebnis. 0, 0.5 und 1 sind alles Vielfache von 0.5. Es können also alle Punkte einer solchen Streichholzanordnung folgendermaßen dargestellt werden: (a*0.5 + b*cos(30) | c*0.5 + d*cos(30)) wobei a bis d ganze Zahlen sind. Da alle Streichhölzer gleich lang sind, kann ein Streichholz als ein Punkt und ein Winkel dargestellt werden.

u ist die Anzahl von Streichhölzern, die umgelegt werden müssen. Wenn beide Streichholzanordnungen übereinandergelegt werden, gibt es Streichhölzer, die dann direkt auf einem Streichholz der anderen Streichholzanordnung liegen und welche, die das nicht tun. Diese werden entsprechend passende und umzulegende Streichhölzer genannt. Wir können die Streichholzanordnungen ineinander überführen, indem alle umzulegenden Streichhölzer der einen Streichholzanordnung auf die der anderen Streichholzanordnung gelegt werden. Beträgt die Differenz von u und der Anzahl umzulegender Streichhölzer 0, das heißt sie sind gleich groß, ist das Rätsel lösbar.

Ist die Differenz größer, das heißt es gibt weniger umzulegende Streichhölzer, ist das Rätsel auch lösbar. Wenn sie größer als 1 ist, werden zusätzlich, der Differenz entsprechend passende Streichhölzer gegenseitig getauscht. So werden u Streichhölzer umgelegt, ohne die

Streichholzanordnung zu verändern.Wenn die Differenz 1 beträgt, wird ein passendes Streichholz umgelegt, anstatt eines der umzulegenden Streichhölzer. Das Ersetzte umzulegende Streichholz wird auf den ehemaligen Platz des passenden Streichholzes gelegt. Wieder wurden ein Streichholz mehr, also u Streichhölzer umgelegt, ohne die Anordnung zu verändern.

Um die beiden Streichholzanordnungen übereinanderzulegen, muss jedes Streichholz einer Streichholzanordnung um einen bestimmten Vektor verschoben werden. Der Algorithmus findet für jedes Rätsel einen Vektor, bei dem das soeben beschriebene Szenario zutrifft bzw. zeigt, dass es diesen Vektor nicht gibt.Nehmen wir die Differenz der Punkte zweier Streichhölzer mit gleichem Winkel von verschiedenen Streichholzanordnungen, erhalten wir einen Vektor, bei dem mindestens ein Streichholz passend ist. Wenn u < n gilt, muss bei dem gesuchten Vektor auch mindestens ein Streichholz passend sein. Der gesuchte Vektor ist also einer dieser Vektoren. Wenn u = n gilt, dann ist das Rätsel auf jeden Fall lösbar, das Szenario trifft also für jeden Vektor zu. Für diesen Fall wird ein zufälliger Vektor ausprobiert.

Danach werden mit allen Streichhölzern der einen und jeweils allen Streichhölzern gleichen Winkels der anderen Streichholzanordnung wie oben beschriebene Vektoren erstellt. Für jeden dieser Vektoren wird geprüft, ob höchstens u Streichhölzer um diesen Vektor verschoben, umzulegend sind. Wenn das zutrifft, dann wurde die Lösung gefunden. Wenn es kein einziges Mal eintrifft, dann ist das Rätsel nicht lösbar.

Optimierungen:

Sobald beim Übereinanderlegen der Streichholzanordnungen die Anzahl an umzulegenden Streichhölzern einer Streichholzanordnung u überschreitet, wird das Aufeinanderlegen frühzeitig abgebrochen. Alle Vektoren, die zu einer Übereinanderlegung genutzt wurden, werden gespeichert. Es werden keine Vektoren mehrmals genutzt.

Umsetzung

Die Lösungsidee wird in Python implementiert. Die Punkte werden als Liste aus 4 Integern dargestellt. Eine Streichholzanordnung ist ein Dictionary mit allen vorkommenden Winkeln als Keys und eine Liste aus allen Punkten von Streichhölzern mit diesem Winkel als Values.

Es werden nur die Winkel von 0 bis 150 verwendet, damit jedes mögliche Streichholz auf genau eine Weise dargestellt werden kann. Ein Streichholz mit Winkel 0 geht dabei in die positive Richtung der x-Achse und Winkel nehmen gegen den Uhrzeigersinn zu.

Die Funktionen “VektorMinus” ermöglicht Vektorsubtraktion, um die zur Verschiebung genutzten Vektoren zu erstellen. Die Funktion “übereinanderlegen” liest einen Vektor ein und verschiebt eine Streichholzanordnung um diesen Vektor. Dann findet sie alle passenden und umzulegenden Streichhölzer der beiden Streichholzanordnung. Wenn die Anzahl der umzulegenden Streichhölzer <= u ist, wird die “ausgeben” Funktion aufgerufen. Diese gibt einen Ergebnissatz zurück.

Die auf den Bwinf-Webseiten zu findenden Streichholzanordnungen werden im Quelltext implementiert. Die Variablennamen setzen sich zusammen aus “Vorher”/”Nachher” je nachdem welche Streichholzanordnung in dem Beispiel überführt werden sollte und der Nummer des Beispiels.

Das Moduls sys wird benutzt, damit das Programm mit den Variablennamen der 2 Streichholzanordnungen und der Anzahl umzulegender Streichhölzer als Parameter auf der Kommandozeile aufgerufen werden kann.

Beispiele

Aus Platzgründen werden nur ein paar Beispiele gezeigt.

$ ./Streichholzrätsel.py Vorher0 Nachher0 3
Das Rätsel ist lösbar.
Die 3 Streichhölzer an folgenden Positionen:
{0: [[0, 0, 0, 0], [0, 0, 2, 0]], 90: [[1, 0, 2, 1]]}
werden an diese Positionen verschoben und gedreht
{60: [[0, 0, 0, 0]], 90: [[1, 0, 0, 1]], 120: [[2, 0, 0, 0]]}.

$ ./Streichholzrätsel.py Vorher1 Nachher1 3
Das Rätsel ist nicht lösbar

$ ./Streichholzrätsel.py Vorher2 Nachher2 4
Das Rätsel ist lösbar.
Die 3 Streichhölzer an folgenden Positionen:
{30: [[1, -1, 1, 1], [5, 0, 0, 1]], 120: [[5, 0, 2, 1]]}
werden an diese Positionen verschoben und gedreht
{30: [[5, -1, 1, 1]], 150: [[5, 0, 0, 1], [1, 0, 2, 1]]}.
[[5, 1, 1, 1], 150] wird an diese Positionen verschoben und gedreht [[5, 0, 2, 1], 120].

$ ./Streichholzrätsel.py Vorher3 Nachher3 10
Das Rätsel ist lösbar.
Die 8 Streichhölzer an folgenden Positionen:
{30: [[0, 0, 0, 0], [0, 3, -1, 0]], 60: [[-1, 2, 0, 1], [0, 2, 0, -2]], 120: [[1, 2, 0, 1], [0, 2, 0, -2]], 150: [[0, 1, -1, 0], [0, 4, 0, 0]]}
werden an diese Positionen verschoben und gedreht
{0: [[0, 2, 0, 0], [-2, 2, 0, 0]], 60: [[-2, 2, 0, -2]], 90: [[0, 2, 0, 0], [0, 2, -2, 0], [0, 2, -4, 0], [0, 2, 2, 0]], 120: [[2, 2, 0, -2]]}.
2 andere Streichhölzer tauschen gegenseitig ihre Plätze.

$ ./Streichholzrätsel.py Vorher4 Nachher4 2
Das Rätsel ist lösbar.
Die 2 Streichhölzer an folgenden Positionen:
{0: [[2, 0, 2, 0], [4, 0, -2, 0]]}
werden an diese Positionen verschoben und gedreht

{0: [[2, 0, -2, 0]], 90: [[2, 0, -2, 0]]}.

Quellcode

```
# Implementierung von Subtraktion von Punkten/Vektoren
def vektorMinus(vektor1, vektor2):
    return [vektor1[index] - vektor2[index] for index in range(0, 4)]


def übereinanderlegen(Vektor):
    def ausgeben(UmzulegendVorher, UmzulegendNachher):
        if u == umzulegend + 1:
            # Wenn es ein umzulegendes Streichholz weniger gibt als u,
            # Wird ein passendes Streichholz statt einem umzulegenden umgelegt
            UmzulegendVorher[passendesStreichholz[1]] += [passendesStreichholz[0]]
            UmzulegendVorher[umzulegendesStreichholz[1]].remove(
                umzulegendesStreichholz[0]
            )
        # UmzulegendVorher und UmzulegendNachher werden gekürzt
        UmzulegendVorher = {k: v for (k, v) in UmzulegendVorher.items() if len(v) > 0}
        UmzulegendNachher = {k: v for (k, v) in UmzulegendNachher.items() if len(v) > 0}
        # Die umzulegenden Streichhölzer von Vorher werden auf die von Nachher gelegt
        Lösung = f"Das Rätsel ist lösbar. Die {umzulegend} Streichhölzer an folgenden Positionen: {
UmzulegendVorher} werden an diese Positionen verschoben und gedreht {UmzulegendNachher}."
        # Die Lösung wird ergänzt, wenn es weniger umzulegende Streichhölzer gibt als u
        if u == umzulegend + 1:
            # Wenn es ein umzulegendes Streichholz weniger gibt als u,
            # Wird das umzulegende Streichholz auf den ehemaligen Platz des passenden gelegt
            Lösung += f"{umzulegendesStreichholz} wird an diese Positionen verschoben und gedreht
{passendesStreichholz}."
        elif u >= umzulegend + 2:
            # Wenn es noch weniger umzulegende Streichhölzer gibt,
            # werden die fehlenden miteinander getauscht
            Lösung += f"{u - umzulegend} andere Streichhölzer tauschen gegenseitig ihre Plätze."
        # Der Lösungssatz Lösung wird zurückgegeben
        return Lösung

    # Jedes Streichholz von Nachher wird um den Vektor verschoben
    # UmzulegendNachher speichert umzulegende Streichhölzer von Nachher
    # UmzulegendVorher speichert umzulegende Streichhölzer von Vorher
    UmzulegendNachher = {
        k: [vektorMinus(i, Vektor) for i in v] for (k, v) in Nachher.items()
    }
    UmzulegendVorher = {0: [], 30: [], 60: [], 90: [], 120: [], 150: []}
    # Mit umzulegend werden die umzulegenden Streichhölzer gezählt,
```

```
    umzulegend = 0
    for Winkel in Vorher:
        for Vorherpunkt in Vorher[Winkel]:
            if Vorherpunkt in UmzulegendNachher[Winkel]:
                # Wenn ein Streichholz passend ist, wird es aus UmzulegendNachher gelöscht,
                # damit am Ende die umzulegenden Streichhölzer übrig bleiben
                UmzulegendNachher[Winkel].remove(Vorherpunkt)
                # passendesStreichholz ist ein passendes Streichholz von Vorher
                passendesStreichholz = [Vorherpunkt, Winkel]
            else:
                # Wenn nicht, wird UmzulegendVorher das Streichholz hinzugefügt
                UmzulegendVorher[Winkel] += [Vorherpunkt]
                umzulegend += 1
                # umzulegendesStreichholz ist ein umzulegendesStreichholz von Vorher
                umzulegendesStreichholz = [Vorherpunkt, Winkel]
        if umzulegend > u:
            # Falls es mehr umzulegende Streichhölzer gibt als u, wird die Funktion abgebrochen
            return None
    # Wenn genau u Streichhölzer umgelegt werden müssen, wird die Lösung zurückgegeben
    return ausgeben(UmzulegendVorher, UmzulegendNachher)


def streichholzrätsel():
    if u == n:
        # Wenn u und n gleich sind, wird jeder Vektor eine Lösung zurückgeben
        return übereinanderlegen([0, 0, 0, 0])
    # Vektoren speichert alle Vektoren
    Vektoren = []
    # Alle Vektoren, bei denen es mindestens 1 umzulegendes Streichholz gibt, werden erstellt
    for Winkel in Vorher:
        for Vorherpunkt in Vorher[Winkel]:
            for Nachherpunkt in Nachher[Winkel]:
                Vektor = vektorMinus(Nachherpunkt, Vorherpunkt)
                # Wenn der Vektor noch nicht getestet wurde,
                # bestimmt die Vergleichsfunktion, ob der Vektor der richtige Vektor ist
                if Vektor not in Vektoren:
                    Vektoren += [Vektor]
                    Ergebnis = übereinanderlegen(Vektor)
                    if Ergebnis:
                        # Wenn eine Lösung gefunden wurde, wird diese zurückgegeben
                        return Ergebnis
    # Wenn keine Lösung gefunden wurde, ist das Rätsel nicht lösbar
    return "Das Rätsel ist nicht lösbar"


print(streichholzrätsel())
```